👢 SPUR: Scaling Reward Learning from Human Demonstrations

Anthony Liang[⋆1], **Yigit Korkmaz**[⋆1], **Jiahui Zhang**[⋆2], **Jesse Zhang**[⋆3], **Abrar Anwar**[1], **Sidhant Kaushik**[4], **Yufei Wang**[5], **Yu Xiang**[2], **David Held**[5], **Dieter Fox**[3], **Abhishek Gupta**[3], **Stephen Tu**[†1], **Erdem Biyik**[†1]

[1]University of Southern California, [2]University of Texas at Dallas, [3]University of Washington, [4]Independent Researcher, [5]Carnegie Mellon University

Abstract

Learning reward functions from human demonstrations is critical for scalable robot learning, yet most approaches either require impractical ground-truth state access, costly online retraining, or yield domain-specific models with poor transferability. We propose SPUR, a unified reward modeling framework that features a large pre-trained vision-language model (VLM) backbone fine-tuned to encode robot image sequences and language instructions, a progress-based reward objective trained on successful demonstrations augmented with rewound videos to simulate failures, and a preference-based learning objective over mismatched and rewound trajectories to enable training on failed executions without explicit progress labels. This design leverages the generalization of VLMs while integrating complementary progress and preference signals for improved robustness and generalization. Experiments on out-of-distribution tasks in simulation show that each component contributes to performance gains across a set of reward metrics, and their combination achieves state-of-the-art results compared to recent baselines, demonstrating scalable training of reward models.

1 Introduction

An important problem in robot learning is that of learning rewards from human demonstrations [33] to guide policy learning. When deploying robots in the real world, it is important that reward models *generalize* to new tasks so that humans will not need to provide additional demonstrations, which is expensive to scale, or train the reward models in tandem with the robot policies, which is sample-inefficient and time-consuming. In this work, we investigate how to train reward functions that can effectively *generalize* to new tasks without online training or additional demonstrations.

Prior works have attempted to develop generalizable reward functions, but they often assume access to ground-truth states, which may be difficult to provide in the real world [22, 17, 43, 28, 29, 24] or the ability to train reward models from scratch in tandem with the policy [33, 38, 41], limiting their practical applicability.

Some recent works instead proposed reward models that can be directly used at test time, conditioned solely on image observations and language instructions. One common approach is to leverage the generalization capabilities of large vision-language models (VLMs) by querying them for *task progress* to be used as reward [36, 27, 2, 13, 30], but these models have been shown to predict noisy rewards, making them difficult to be directly used for training robot policies [2, 13, 44]. Another is to directly train a smaller reward model on human demonstrations. These methods use either a task-progress-based training objective [26, 18, 44], or a preference-based or contrastive objective [40, 3, 19], but they result in domain-specific reward models that are unlikely to generalize

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Embodied World Models for Decision Making.

well to new domains. Instead, we aim to train a generalizable reward model that can provide useful rewards, even on significantly out-of-distribution tasks and settings. We hypothesize that ideas from all three threads of work are useful, and unifying them into a single framework can lead to a reward model with greater generalization capability.

To this end, we investigate how to blend together large-scale VLM backbones, progress-based rewards, and preference-based rewards into one scalable, unified reward model we call SPUR (**S**calable **P**rogress and Preference **U**nified **R**eward). Firstly, we investigate the use of a large-scale pre-trained VLM backbone, not for zero-shot robot reward queries, but instead as a trainable backbone for encoding robot image sequences and language instruction tokens. SPUR then directly predicts task progress coming from successful demonstration trajectories, along with simulating failed trajectories with *video rewind* augmentation [44], to produce useful per-timestep rewards for robots. Finally, to help the model scale, SPUR also trains to predict binary *preferences* over mismatched and rewound video sequences. This preference objective complements the reward prediction objective while also allowing for training with trajectories with *failed execution*, which progress-based methods cannot train on without explicit progress labels for each failed trajectory.

Through reward analysis experiments on new tasks in LIBERO [25] and Meta-World [42], we demonstrate how each component complements the others for scalable training of generalizable reward models. SPUR outperforms recent, state-of-the-art baselines across metrics in both domains.

2 Related Works

2.1 Learning Reward Functions

Several prior works explored learning reward functions from various forms of supervision. One line of research leverages direct human feedback, such as comparisons [7, 35, 6, 23, 15], rankings [32], language annotations [41], and trajectory corrections [21, 5], to infer rewards. While these methods can align reward functions with human intent, they typically require substantial human supervision and are often sample-inefficient.

Another major direction is inverse RL (IRL), where reward functions are inferred from demonstrations [33, 1, 45, 10] or implicitly from expert and goal-state distributions [16, 11, 12]. However, IRL methods struggle to scale to high-dimensional state-action spaces and usually require new demonstrations for every new task. In general, both human-feedback-based and IRL-based approaches lack effective transfer mechanisms: when faced with a novel task, they often need to be retrained from scratch. In contrast, SPUR leverages the semantic representations in VLM backbone to transfer learned reward functions to unseen tasks without requiring additional human supervision.

2.2 Large Vision and Language Models as Reward Functions

Recently, LLMs and VLMs have been applied to reward design through code generation [28, 43, 39], embedding-based reward estimation [31, 36], and preference-based feedback [38, 22]. However, most of these methods assume access to privileged state information that is rarely available in real-world settings. Another line of work employs VLMs as zero-shot success detectors, treating them as sparse reward models [34, 9, 14]. While promising, this approach provides only episodic feedback and misses the dense supervision signals present throughout the trajectory.

Some prior work explores task progress as a proxy reward, either by using VLMs as progress estimators [36, 27, 2, 13, 30] or by training task-specific models with progress-prediction objectives [26, 18, 44]. VLM-based estimators, however, often yield noisy outputs, while smaller per-task models tend to overfit to domain-specific dynamics, limiting their generalization to new domains. In this work, we combine progress prediction with preference feedback over video sequences to improve the reward learning objective. We further show that incorporating failure trajectory pairs improves generalization across tasks.

3 Method

We introduce SPUR, a generalizable reward model, as illustrated in Figure 1. We start with a dataset $\mathcal{D} = \{\tau_1, \tau_2, \tau_3, ...\}$ consisting of robot demonstration trajectories $\tau = \{o_{1:T}, l, \texttt{success}\}$ with

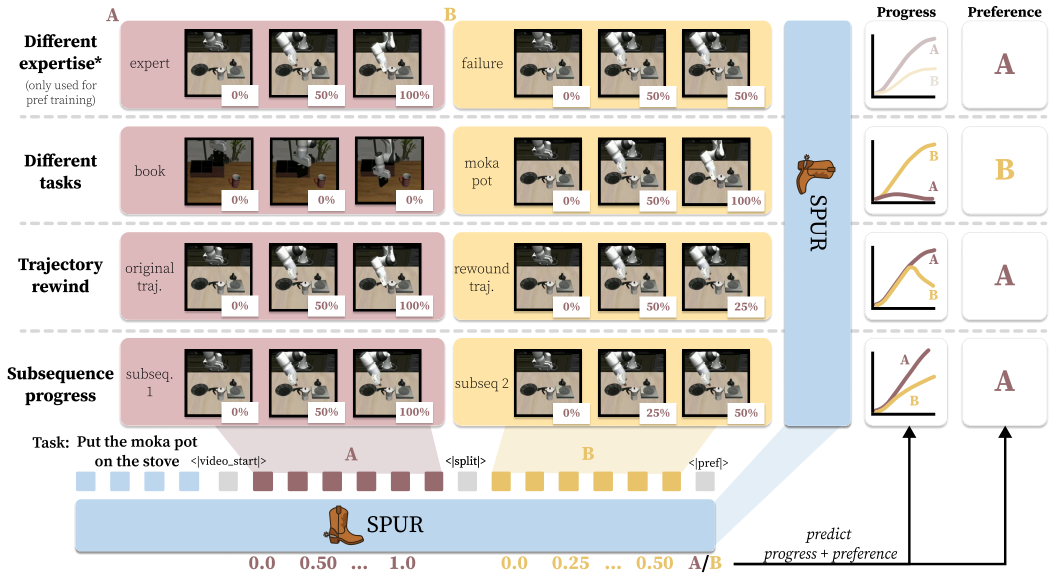

Figure 1: **SPUR.** Given two video trajectories, we train our VLM-based reward model, SPUR, to predict progress-based and preference-based rewards. We use four strategies (left) for curating training examples from our given datasets, which are further detailed in Section 3.2.

image observations o, language instructions l, and a success label $\texttt{success} \in \{0, 1\}$. To enable generalization to unseen tasks, environments, and domains, we first instantiate the reward model with a large-scale, pre-trained vision-language model (VLM) backbone. Then, we fine-tune it on two objectives that complement each other: predicting *preferences* over pairs of video trajectories and predicting continuous task *progress* as rewards.

3.1 VLM Base Model

Our base model is QWEN2.5-VL-INSTRUCT-3B [4], a 3B parameter, open-source, image and video-input VLM which demonstrates strong zero-shot performance across various vision and language tasks. SPUR can incorporate any base VLM model which supports language and video input, but we found QWEN to be easy to tune and performant. SPUR uses this model to take as input a natural language task description l and up to two different video sequences, $o^1_{1:T}$ and $o^2_{1:T}$ of arbitrary length. SPUR encodes both the language and videos as a single sequence of tokens with the base model's tokenizer to construct its inputs as depicted below:

$$(l, o^1, o^2) \rightarrow \texttt{Token}(l)\ \langle|\text{video\_start}|\rangle\ \texttt{Token}(o^1)\ \langle|\text{split\_token}|\rangle\ \texttt{Token}(o^2)\ \langle|\text{pref\_token}|\rangle, \tag{1}$$

where $\langle|\text{split\_token}|\rangle$ is a special token that delinates the two video sequences. The VLM then produces a sequence of hidden states, which we use for preference and progress prediction, as detailed next.

3.2 Preference Prediction

To predict preferences, we attach an MLP head to the final hidden state corresponding to the special token $\langle|\text{pref\_token}|\rangle$ from Equation (1) to produce preference logits. The model is trained to discern which of the two video sequences, $o^1_{1:T}$ or $o^2_{1:T}$, is better aligned with the given natural language task description, l. We denote the preference label as y, where $y = 1$ if o^1 is preferred over o^2, and $y = 0$ otherwise. Formally, the learned preference head MLP_{pref} produces a probability:

$$P(o^1 \succ o^2 \mid l) = \sigma\left(\text{MLP}_{\text{pref}}\left(h_{\langle|\text{pref\_token}|\rangle}\right)\right).$$

where σ is the sigmoid function and $h_{\langle|\text{pref\_token}|\rangle}$ is the hidden state corresponding to the location of the $\langle|\text{pref\_token}|\rangle$ in the input from Equation (1). The preference objective is optimized using the binary cross-entropy loss and backpropagated through MLP_{pref} and the VLM through $h_{\langle|\text{pref\_token}|\rangle}$:

$$\mathcal{L}_{\text{preference}} = -\Big[y \log P(o^1 \succ o^2 \mid l) + (1-y)\log(1 - P(o^1 \succ o^2 \mid l))\Big].$$

Preference Sample Construction. Large-scale preference datasets comparing robot trajectories are not widely available, especially for training generalizable reward models. Given the scarcity of such data, we instead propose a suite of strategies for scalably curating a larger set of preference samples from existing trajectories without needing manual human annotations.

We construct preference pairs $(l, o^{\text{chosen}}, o^{\text{rejected}}, y)$ for training by sampling trajectories from $\mathcal{D}$, always assigning o^{chosen} as the preferred observation sequence ($y = 1$). Given sampled trajectories $\tau = \{o_{1:T}, l, \texttt{success}\}$, we create batches of preference tuples sampled uniformly over the following four strategies:

1. **Different expertise.** Given a task instruction l, sample two trajectories $\tau_1, \tau_2 \sim \mathcal{D}$ with the same instruction where τ_1 has $\texttt{success} == 1$ and τ_2 has $\texttt{success} == 0$. We extract o^{chosen} from the observation sequence from τ_1.
2. **Different tasks.** Sample a trajectory $o^{\text{chosen}} \sim \mathcal{D}$ corresponding to the task instruction l and a trajectory o^{rejected} with a different instruction. These samples encourage the model to ground correct video and language pairs.
3. **Trajectory rewind.** Following the idea proposed by ReWiND [44] that generated failed trajectories for reward *progress* prediction by *rewinding* videos, we propose to rewind successful videos to generate negative preference pairs. For a given trajectory $o^{\text{chosen}} = o_{1:T}$ with $\texttt{success} == 1$, we first sample a random contiguous subsegment:

$$o_{\text{sub}} = o_{1:t_{\text{end}}}, \quad 1 \le t_{\text{end}} \le T.$$

   We then generate a *rewound* trajectory o^{rejected} by reversing the last k frames of the o_{sub} where $k \sim \mathcal{U}(1, t_{\text{end}} - t_{\text{start}})$:

$$o^{\text{rejected}} = [o_{1:t_{\text{end}}}, o_{t_{\text{end}-1}:t_{\text{end}-k+1}}],$$

   where $[\cdot]$ denotes concatenating the videos. This procedure ensures that o^{chosen} represents the full progress along the subsegment, while o^{rejected} exhibits backward progress at the end.
4. **Subsequence progress.** For the same trajectory τ with $\texttt{success} == 1$, sample two subsequences $o_{1:t_1}, o_{1:t_2}$ with $t_1 < t_2$. We assign $o^{\text{chosen}} = o_{1:t_2}$ as it is further along in the task.

In practice, for all of these samples, we also sample the first frame randomly from the first half of the trajectory so that in datasets where the robot's starting position is consistent across trajectories, SPUR does not overfit to the robot's starting position.

3.3 Task Progress Prediction

In addition to preference prediction, SPUR also predicts the per-frame *progress* for each video as it can more directly be used for rewarding policies downstream [44]. Given a video $o_{1:T}$ with language instruction l, SPUR predicts a continuous progress value $p \in [0, 1]$ indicating the fraction of the task completed at each frame. The tokenized prompt is the same as in Equation (1) except without the second video o^2.

Specifically, a progress prediction MLP head, $\text{MLP}_{\text{progress}}$, is attached to the hidden states $h_{\langle|o_i|\rangle}$ corresponding to each frame i, thereby producing per-frame progress predictions. We train SPUR on the same data as in Section 3.2, with the exception of "Different expertise" where failed trajectories are not used for progress training as they do not have a ground truth progress to use. For a given video from a sampled trajectory $o_{1:T}$ (which can also be a subsequence), the progress prediction loss is computed as the Mean Squared Error (MSE) between predicted and ground-truth progress values:

$$
\mathcal{L}_{\text{progress}} = \begin{cases} \sum_{t=1}^{T} \left(\text{MLP}_{\text{progress}}(h_{\langle|o_t|\rangle}) - \underbrace{t/T}_{\text{ground truth progress}} \right)^2, & \text{if not rewound} \\ \sum_{t=1}^{T} \left(\underbrace{\text{MLP}_{\text{progress}}(h_{\langle|o_t|\rangle}) - 0}_{\text{0 progress for mismatched tasks}} \right)^2 & \text{if wrong task} \\ \underbrace{\sum_{t=1}^{t_{\text{end}}} \left(\text{MLP}_{\text{progress}}(h_{\langle|o_t|\rangle}) - \frac{t}{T} \right)^2}_{\text{Loss for original trajectory until } t_{\text{end}}} + \underbrace{\sum_{t=1}^{k} \left(\text{MLP}_{\text{progress}}(h_{\langle|o_t|\rangle}) - \frac{t_{\text{end}} - t}{T} \right)^2}_{\text{Rewound video for } k \text{ frames from } t_{\text{end}}-1}, & \text{if rewound.} \end{cases} \quad (2)
$$

We compute progress losses only for `success` trajectories, ensuring that the model learns meaningful temporal progress where the task is at least partially completed.

Overall, our final pretraining objective for SPUR is: $\mathcal{L}_{\text{preference}} + \mathcal{L}_{\text{progress}}$.

4 Experiments

Our experiments aim to study the efficacy of each component of SPUR and compare it against baselines across a wide array of reward metrics. To this end, we organize our experiments to answer the following experimental questions, in order:

(**Q1**) Which components of SPUR contribute the most to generalizable reward prediction?
(**Q2**) How does SPUR compare against baselines across a variety of reward metrics in unseen tasks?

Setup: We conduct experiments using the **LIBERO-90** dataset from the Lifelong Robot Learning Suite [25]. This dataset provides a diverse set of household manipulation tasks with various levels of distribution shift. Models are trained on demonstrations for 90 tasks in LIBERO-90 and evaluated on four benchmark splits: **LIBERO-10**, **Object**, **Spatial**, and **Goal**, which measure generalization across different dimensions such as goal, object, and spatial configurations. The original benchmark includes 4500 trajectories (50 per task) rendered at 128x128; following Kim et al. [20], we replay and re-render them at 256x256 and discard trajectories that did not replay successfully. We also include a corresponding set of failed trajectories constructed by replaying demonstration trajectories with added Gaussian noise on the actions.

We additionally compare on **MetaWorld** [42], specifically the 20-task training split consisting of 5 demonstrations each from Zhang et al. [44]. Correspondingly, we evaluate on the corresponding 17-task evaluation dataset across a variety of metrics proposed by Zhang et al. [44] that were shown to be reflective of downstream policy performance.

We list all dataset sizes in Table 4.

Baselines: We compare SPUR against several strong reward learning baselines:

- **ReWiND** [44] trains a transformer-based network with a direct progress prediction objective using frozen language and image encoders along with video rewinding to simulate failed policy rollouts.
- **Generative Value Learning (GVL)** [30] prompts a pre-trained Gemini LLM [37] with shuffled video frames to predict task progress for subsampled frames across the video sequence. We also convert its progress predictions to preference predictions by comparing last-frame predicted task progress between queried trajectories.
- **RL-VLM-F** [38] prompts a pre-trained LLM to obtain preference-based feedback predictions. We query Gemini for these preference predictions.

4.1 Q1: Which Components of SPUR Contribute the Most?

First, we ablate individual components of SPUR to measure the effect of each. For these experiments, we train exclusively on LIBERO-90 data (both success and failure) and evaluate on the unseen LIBERO-10, Object, Spatial, and Goal datasets.

Table 1: **LIBERO Ablation Analysis.** Comparison of ablations across preference and progress accuracy metrics across unseen tasks in LIBERO-10, Object, Spatial, and Goal after training on LIBERO-90. – indicates metrics that are not applicable to the given model.

Category	Metric	Base Model	w/o Pref.	w/o Progress	w/o Fail. Traj.	SPUR
Preference Accuracy	Failed Trajs. ↑	0.5	0.64	0.82	0.69	**0.91**
Progress Accuracy	MSE ↓	–	0.04	–	0.04	**0.03**
	Reward Alignment $\rho\uparrow$	–	0.73	–	0.73	**0.81**

- **Base Model**: Uses the pre-trained QWEN-2.5-VL-INSTRUCT-3B model to produce preference and progress predictions via direct text prompting.
- **w/o Preference**: Removes preference losses from the training objective. Preference accuracy is computed by using final-frame progress comparisons instead.
- **w/o Progress**: Removes progress losses from the training objective.
- **w/o Failure Data**: Removes unsuccessful trajectories from the training objective.

Reward Metrics. We compute: **preference accuracy** when comparing paired successful and failed trajectories, and **progress prediction accuracy** in terms of mean-squared-error (MSE) against the ground-truth progress target of successful trajectories and in terms of reward *alignment* in terms of spearman correlation (ρ), measuring how well the predicted progress is ordered with respect to the ground truth progress ordering of successful demonstrations.

Results averaged across our 4 unseen task distributions are displayed in Table 1, where the base model performs at random chance on predicting preferences. We found it almost always produced deterministically increasing progress predictions, so we do not include progress accuracy metrics. Meanwhile, removing preference predictions hurts the progress accuracy and reward alignment compared to SPUR, and removing progress predictions hurts the preference accuracy relative to SPUR. Removing failed trajectories also predictably hurts unseen failed trajectory preference accuracy. Overall, we demonstrate that SPUR performs the best across all comparisons and that each component we ablate complements each other to increase overall performance.

4.2 Q2: Reward Function Analysis in Unseen Tasks

Table 2: **LIBERO Metrics.** Baseline comparison across preference and progress accuracy metrics across unseen tasks in LIBERO-10, Object, Spatial, and Goal after training on LIBERO-90.

Category	Metric	RL-VLM-F	GVL	SPUR
Preference Accuracy	Failed Trajs.	0.39	0.65	**0.91**
Progress Accuracy	MSE ↓	–	0.07	**0.03**
	Reward Alignment $\rho\uparrow$	–	0.68	**0.81**

Now, we compare SPUR against reward model baselines across unseen tasks in both LIBERO and Metaworld. We first list **LIBERO** comparisons in Table 2 to GVL and Rl-VLM-F. All methods are trained on the same LIBERO-90 datasets where applicable (GVL and RL-VLM-F instead prompt pre-trained, closed-source generative models). We can see that SPUR outperforms RL-VLM-F by **2.9x** and GVL by **1.4x** on preference accuracy. Additionally, it outperforms GVL with less than half the progress prediction MSE and **1.19x** improvement on reward alignment correlation.

Table 3: **Meta-World Reward Metrics.** Comparison of reward models in terms of reward alignment (ρ) on Meta-World. Baseline results taken from ReWiND [44].

Category	Metric	LIV-FT	RoboCLIP	VLC	GVL	ReWiND w/o OXE	ReWiND w/ OXE	SPUR
Reward Alignment	$\rho\uparrow$	0.55	-0.01	0.62	0.57	0.64	0.79	**0.83**

Next we compare **Meta-World** performance against an additional set of baselines on the Meta-World evaluation dataset from ReWiND [44]. For a more comprehensive comparison, we also include additional baselines listed in Zhang et al. [44], namely LIV-FT [26], VLC [3], and RoboCLIP [36], along with ReWiND trained with and without the Open X-Embodiment (OXE) Dataset [8] as proposed by Zhang et al. [44]. Results in Table 3 indicate that SPUR outperforms the best-performing model, beating ReWiND even when it is trained with additional data from OXE, and beating ReWiND's performance by **1.29x** when both models are trained on the same data (w/o OXE).

5 Conclusion

We studied the problem of learning reward functions that generalize to unseen tasks without relying on additional demonstrations or online training. To address these challenges, we introduced SPUR, a unified reward learning framework that leverages a large-scale VLM backbone together with both progress-based and preference-based objectives. By combining per-timestep progress prediction with preference supervision over mismatched and rewound trajectories, SPUR learns from both successful and failed executions while producing denser and more transferable rewards. Our experiments on LIBERO and Meta-World show that each component of SPUR contributes to improved generalization, and that the full model consistently outperforms recent state-of-the-art baselines across diverse reward metrics.

Looking forward, we believe that scalable reward learning frameworks such as SPUR offer a promising path toward reducing reliance on costly demonstrations and enabling more robust robot policy training in real-world settings. Future directions include extending our framework to longer-horizon tasks, enabling cross-embodiment reward transfer including human videos, and evaluating deployment in real-robot experiments.

References

[1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2004.

[2] A. Adeniji, A. Xie, C. Sferrazza, Y. Seo, S. James, and P. Abbeel. Language reward modulation for pretraining reinforcement learning. In *In RLC Reinforcement Learning Beyond Rewards Workshop 2024*, 2024. URL https://arxiv.org/abs/2308.12270.

[3] M. Alakuijala, R. McLean, I. Woungang, N. Farsad, S. Kaski, P. Marttinen, and K. Yuan. Video-language critic: Transferable reward functions for language-conditioned robotics. In *Transactions on Machine Learning Research (TMLR)*, 2025.

[4] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[5] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan. Learning from physical human corrections, one feature at a time. In *International Conference on Human-Robot Interaction (HRI)*, 2018.

[6] E. Biyik, N. Huynh, M. J. Kochenderfer, and D. Sadigh. Active preference-based gaussian process regression for reward learning. In *Robotics: Science and Systems (RSS)*, 2020.

[7] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In *NeurIPS*, 2017.

[8] O. X.-E. Collaboration, A. O'Neill, A. Rehman, A. Gupta, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, A. Tung, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Gupta, A. Wang, A. Kolobov, A. Singh, A. Garg, A. Kembhavi, A. Xie, A. Brohan, A. Raffin, A. Sharma, A. Yavary, A. Jain, A. Balakrishna, A. Wahid, B. Burgess-Limerick, B. Kim, B. Schölkopf, B. Wulfe, B. Ichter, C. Lu, C. Xu, C. Le, C. Finn, C. Wang, C. Xu, C. Chi, C. Huang, C. Chan, C. Agia, C. Pan, C. Fu, C. Devin, D. Xu, D. Morton, D. Driess, D. Chen, D. Pathak, D. Shah, D. Büchler, D. Jayaraman, D. Kalashnikov, D. Sadigh, E. Johns, E. Foster, F. Liu, F. Ceola, F. Xia, F. Zhao, F. V. Frujeri, F. Stulp, G. Zhou, G. S. Sukhatme, G. Salhotra, G. Yan, G. Feng, G. Schiavi, G. Berseth, G. Kahn, G. Yang, G. Wang, H. Su, H.-S. Fang, H. Shi, H. Bao, H. B. Amor, H. I. Christensen, H. Furuta, H. Bharadhwaj, H. Walke, H. Fang, H. Ha, I. Mordatch, I. Radosavovic, I. Leal, J. Liang, J. Abou-Chakra, J. Kim, J. Drake, J. Peters, J. Schneider, J. Hsu, J. Vakil, J. Bohg, J. Bingham, J. Wu, J. Gao,

J. Hu, J. Wu, J. Wu, J. Sun, J. Luo, J. Gu, J. Tan, J. Oh, J. Wu, J. Lu, J. Yang, J. Malik, J. Silvério, J. Hejna, J. Booher, J. Tompson, J. Yang, J. Salvador, J. J. Lim, J. Han, K. Wang, K. Rao, K. Pertsch, K. Hausman, K. Go, K. Gopalakrishnan, K. Goldberg, K. Byrne, K. Oslund, K. Kawaharazuka, K. Black, K. Lin, K. Zhang, K. Ehsani, K. Lekkala, K. Ellis, K. Rana, K. Srinivasan, K. Fang, K. P. Singh, K.-H. Zeng, K. Hatch, K. Hsu, L. Itti, L. Y. Chen, L. Pinto, L. Fei-Fei, L. Tan, L. J. Fan, L. Ott, L. Lee, L. Weihs, M. Chen, M. Lepert, M. Memmel, M. Tomizuka, M. Itkina, M. G. Castro, M. Spero, M. Du, M. Ahn, M. C. Yip, M. Zhang, M. Ding, M. Heo, M. K. Srirama, M. Sharma, M. J. Kim, N. Kanazawa, N. Hansen, N. Heess, N. J. Joshi, N. Suenderhauf, N. Liu, N. D. Palo, N. M. M. Shafiullah, O. Mees, O. Kroemer, O. Bastani, P. R. Sanketi, P. T. Miller, P. Yin, P. Wohlhart, P. Xu, P. D. Fagan, P. Mitrano, P. Sermanet, P. Abbeel, P. Sundaresan, Q. Chen, Q. Vuong, R. Rafailov, R. Tian, R. Doshi, R. Mart'in-Mart'in, R. Baijal, R. Scalise, R. Hendrix, R. Lin, R. Qian, R. Zhang, R. Mendonca, R. Shah, R. Hoque, R. Julian, S. Bustamante, S. Kirmani, S. Levine, S. Lin, S. Moore, S. Bahl, S. Dass, S. Sonawani, S. Tulsiani, S. Song, S. Xu, S. Haldar, S. Karamcheti, S. Adebola, S. Guist, S. Nasiriany, S. Schaal, S. Welker, S. Tian, S. Ramamoorthy, S. Dasari, S. Belkhale, S. Park, S. Nair, S. Mirchandani, T. Osa, T. Gupta, T. Harada, T. Matsushima, T. Xiao, T. Kollar, T. Yu, T. Ding, T. Davchev, T. Z. Zhao, T. Armstrong, T. Darrell, T. Chung, V. Jain, V. Kumar, V. Vanhoucke, W. Zhan, W. Zhou, W. Burgard, X. Chen, X. Chen, X. Wang, X. Zhu, X. Geng, X. Liu, X. Liangwei, X. Li, Y. Pang, Y. Lu, Y. J. Ma, Y. Kim, Y. Chebotar, Y. Zhou, Y. Zhu, Y. Wu, Y. Xu, Y. Wang, Y. Bisk, Y. Dou, Y. Cho, Y. Lee, Y. Cui, Y. Cao, Y.-H. Wu, Y. Tang, Y. Zhu, Y. Zhang, Y. Jiang, Y. Li, Y. Li, Y. Iwasawa, Y. Matsuo, Z. Ma, Z. Xu, Z. J. Cui, Z. Zhang, Z. Fu, and Z. Lin. Open X-Embodiment: Robotic learning datasets and RT-X models. In *International Conference on Robotics and Automation (ICRA)*, 2024.

[9] Y. Du, K. Konyushkova, M. Denil, A. Raju, J. Landon, F. Hill, N. de Freitas, and S. Cabi. Vision-language models as success detectors. In *Proceedings of The 2nd Conference on Lifelong Learning Agents*, pages 120–136, 2023.

[10] C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International Conference on Machine Learning (ICML)*, 2016.

[11] J. Fu, K. Luo, and S. Levine. Learning robust rewards with adversarial inverse reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2018.

[12] J. Fu, A. Singh, D. Ghosh, L. Yang, and S. Levine. Variational inverse control with events: A general framework for data-driven reward definition. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *NeurIPS*, 2018.

[13] Y. Fu, H. Zhang, D. Wu, W. Xu, and B. Boulet. FuRL: Visual-language models as fuzzy rewards for reinforcement learning. In *Internatonal Conference on Machine Learning*, 2024.

[14] L. Guan, Y. Zhou, D. Liu, Y. Zha, H. B. Amor, and S. Kambhampati. Task success is not enough: Investigating the use of video-language models as behavior critics for catching undesirable agent behaviors. In *Conference on Language Modeling*, 2024.

[15] J. Hejna and D. Sadigh. Few-shot preference learning for human-in-the-loop rl. In *Conference on Robot Learning (CoRL)*, 2022.

[16] J. Ho and S. Ermon. Generative adversarial imitation learning. In *NeurIPS*, 2016.

[17] H. Hu and D. Sadigh. Language instructed reinforcement learning for human-ai coordination. In *International Conference on Machine Learning (ICML)*, 2023.

[18] K.-H. Hung, P.-C. Lo, J.-F. Yeh, H.-Y. Hsu, Y.-T. Chen, and W. H. Hsu. VICtor: Learning hierarchical vision-instruction correlation rewards for long-horizon manipulation. In *International Conference on Learning Representations (ICLR)*, 2025.

[19] C. Kim, M. Heo, D. Lee, H. Lee, J. Shin, J. J. Lim, and K. Lee. Subtask-aware visual reward learning from segmented demonstrations. In *International Conference on Learning Representations (ICLR)*, 2025.

[20] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. OpenVLA: An open-source vision-language-action model. In *Conference on Robot Learning (CoRL)*, 2024.

[21] Y. Korkmaz and E. Bıyık. Mile: Model-based intervention learning. In *International Conference on Robotics and Automation (ICRA)*, 2025.

[22] M. Kwon, S. M. Xie, K. Bullard, and D. Sadigh. Reward design with language models. In *International Conference on Learning Representations (ICLR)*, 2023.

[23] K. Lee, L. Smith, and P. Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In *International Conference on Machine Learning (ICML)*, 2021.

[24] W. Liang, S. Wang, H.-J. Wang, O. Bastani, D. Jayaraman, and Y. J. Ma. Environment curriculum generation via large language models. In *Conference on Robot Learning (CoRL)*, 2024.

[25] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023.

[26] Y. J. Ma, W. Liang, V. Som, V. Kumar, A. Zhang, O. Bastani, and D. Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning (ICML)*, 2023.

[27] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. In *International Conference on Learning Representations (ICLR)*, 2023.

[28] Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar. Eureka: Human-level reward design via coding large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

[29] Y. J. Ma, W. Liang, H. Wang, S. Wang, Y. Zhu, L. Fan, O. Bastani, and D. Jayaraman. Dreureka: Language model guided sim-to-real transfer. In *Robotics: Science and Systems (RSS)*, 2024.

[30] Y. J. Ma, J. Hejna, A. Wahid, C. Fu, D. Shah, J. Liang, Z. Xu, S. Kirmani, P. Xu, D. Driess, T. Xiao, J. Tompson, O. Bastani, D. Jayaraman, W. Yu, T. Zhang, D. Sadigh, and F. Xia. Vision language models are in-context value learners. In *International Conference on Learning Representations (ICLR)*, 2025.

[31] P. Mahmoudieh, D. Pathak, and T. Darrell. Zero-shot reward specification via grounded natural language. In *International Conference on Machine Learning (ICML)*, 2022.

[32] V. Myers, E. Biyik, N. Anari, and D. Sadigh. Learning multimodal rewards from rankings. In *Conference on Robot Learning (CoRL)*, 2021.

[33] A. Y. Ng and S. J. Russell. Algorithms for inverse reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2000.

[34] J. Rocamonde, V. Montesinos, E. Nava, E. Perez, and D. Lindner. Vision-language models are zero-shot reward models for reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2024.

[35] D. Sadigh, A. D. Dragan, S. S. Sastry, and S. A. Seshia. Active preference-based learning of reward functions. In *Robotics: Science and Systems (RSS)*, 2017.

[36] S. A. Sontakke, J. Zhang, S. Arnold, K. Pertsch, E. Biyik, D. Sadigh, C. Finn, and L. Itti. Roboclip: One demonstration is enough to learn robot policies. In *NeurIPS*, 2023.

[37] G. Team. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2024.

[38] Y. Wang, Z. Sun, J. Zhang, Z. Xian, E. Biyik, D. Held, and Z. Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. In *International Conference on Machine Learning (ICML)*, 2024.

[39] T. Xie, S. Zhao, C. H. Wu, Y. Liu, Q. Luo, V. Zhong, Y. Yang, and T. Yu. Text2reward: Reward shaping with language models for reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2024.

[40] D. Yang, D. Tjia, J. Berg, D. Damen, P. Agrawal, and A. Gupta. Rank2reward: Learning shaped reward functions from passive video. In *International Conference on Robotics and Automation (ICRA)*, 2024.

[41] Z. Yang, M. Jun, J. Tien, S. J. Russell, A. Dragan, and E. Bıyık. Trajectory improvement and reward learning from comparative language feedback. In *Conference on Robot Learning (CoRL)*, 2024.

[42] T. Yu, D. Quillen, Z. He, R. Julian, A. Narayan, H. Shively, A. Bellathur, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019.

[43] W. Yu, N. Gileadi, C. Fu, S. Kirmani, K.-H. Lee, M. Gonzalez Arenas, H.-T. Lewis Chiang, T. Erez, L. Hasenclever, J. Humplik, B. Ichter, T. Xiao, P. Xu, A. Zeng, T. Zhang, N. Heess, D. Sadigh, J. Tan, Y. Tassa, and F. Xia. Language to rewards for robotic skill synthesis. In *Conference on Robot Learning (CoRL)*, 2023.

[44] J. Zhang, Y. Luo, A. Anwar, S. A. Sontakke, J. J. Lim, J. Thomason, E. Biyik, and J. Zhang. ReWiND: Language-guided rewards teach robot policies without new demonstrations. In *9th Annual Conference on Robot Learning*, 2025. URL https://openreview.net/forum?id=XjjXLxfPou.

[45] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2008.

A Impact Statement

This paper introduces a unified framework for learning generalizable reward functions by combining vision–language model backbones with progress- and preference-based objectives. Our approach reduces reliance on costly demonstrations and improves transfer to unseen tasks, making robot learning more scalable. Nonetheless, it inherits limitations of large pretrained models, including potential bias and limited interpretability, and thus requires additional safeguards for safe real-world deployment.

B Dataset Specs and Training Configuration

Table 4: Dataset

Dataset Splits	
Dataset	**Num Trajectories**
LIBERO90	3950
LIBERO10	388
LIBERO-Goal	432
LIBERO-Spatial	433
LIBERO-Object	456
LIBERO90 Failure	4312
LIBERO10 Failure	498
MetaWorld Train	100
MetaWorld Eval	85

Table 5: Configuration Parameters for SPUR Training

Training Configuration for RFM	
Parameter	**Value**
Base Model	`Qwen/Qwen2.5-VL-3B-Instruct`
Max frames (downsampled)	16
Per device training batch size	16
Learning rate	2e-5
Training steps	5000
Max sequence length	1024
LR scheduler	Cosine
Warmup ratio	0.1
Expertise / Task / Rewind / Subsequence ratio	[0.3, 0.3, 0.4, 0.0]

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Paper's contributions and scope are summarized in detail in the abstract and the introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The authors explain the limitations of the proposed work at the end of the paper, in the conclusion section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All the details required to reproduce the results are provided in the main paper and the supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The authors will release their code with sufficient instructions reproduce the experiments.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The training and test details to understand and reproduce the results are provided in the main paper and in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All of the results presented in the paper provide information about the statistical significance of the experiments with plots including standard deviation across runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The provides information about the type of compute workers CPU or GPU, internal cluster used for running the experiments in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: In the supplementary.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The authors stated which version of the asset is used and cited the original papers that produced the code package.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: NA

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.