SkillDiffuser: Interpretable Hierarchical Planning via Skill Abstractions in Diffusion-Based Task Execution

Supplementary Material

A. Theoretical Foundation of Classifier-free Diffusion Model for Planning

A.1. Review of Classifier-guided Diffusion Model

Firstly, for a given trajectory $\boldsymbol{\tau}$, the standard reverse process of an unconditional diffusion probabilistic model is defined by $p_\theta(\boldsymbol{\tau}^i|\boldsymbol{\tau}^{i+1})$. This framework is then extended to incorporate conditioning on a specific label $\boldsymbol{y}$ (*e.g.*, the reward), which is considered in the context of current-step denoised trajectory $\boldsymbol{\tau}^i$, which is represented as $p_\phi(\boldsymbol{y}|\boldsymbol{\tau}^i)$. Consequently, the reverse diffusion process can be reformulated as $p_{\theta,\phi}(\boldsymbol{\tau}^i|\boldsymbol{\tau}^{i+1},\boldsymbol{y})$. This approach introduces additional parameters ϕ alongside the original diffusion model parameters θ. The parameters ϕ can be viewed as a classifier, that encapsulates the probability of whether a noisy trajectory $\boldsymbol{\tau}^i$ satisfies the specific label $\boldsymbol{y}$, with a notation of $p_\phi(\boldsymbol{y}|\boldsymbol{\tau}^i)$.

Under the constraints illustrated in [6, 23], we can derive the following theorem with lemma

$$p_{\theta,\phi}\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i, \boldsymbol{\tau}^{i+1}\right) = p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right). \tag{12}$$

Theorem A.1. *The conditional sampling probability of reverse diffusion process $p_{\theta,\phi}(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1},\boldsymbol{y})$ is proportional to unconditional transition probability $p_\theta(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1})$ multiplied by the classified probability $p_\phi(\boldsymbol{y} \mid \boldsymbol{\tau}^i)$.*

$$p_{\theta,\phi}(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1},\boldsymbol{y}) = Z p_\theta(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1}) p_\phi(\boldsymbol{y} \mid \boldsymbol{\tau}^i) \tag{13}$$

Proof.

$$\begin{aligned}
p_{\theta,\phi}(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1},\boldsymbol{y}) &= \frac{p_{\theta,\phi}\left(\boldsymbol{\tau}^i,\boldsymbol{\tau}^{i+1},\boldsymbol{y}\right)}{p_{\theta,\phi}\left(\boldsymbol{\tau}^{i+1},\boldsymbol{y}\right)} \\
&= \frac{p_{\theta,\phi}\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i,\boldsymbol{\tau}^{i+1}\right) p_\theta\left(\boldsymbol{\tau}^i,\boldsymbol{\tau}^{i+1}\right)}{p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^{i+1}\right) p_\theta\left(\boldsymbol{\tau}^{i+1}\right)} \\
&= \frac{p_{\theta,\phi}\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i,\boldsymbol{\tau}^{i+1}\right) p_\theta\left(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1}\right) p_\theta\left(\boldsymbol{\tau}^{i+1}\right)}{p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^{i+1}\right) p_\theta\left(\boldsymbol{\tau}^{i+1}\right)} \\
&= \frac{p_{\theta,\phi}\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i,\boldsymbol{\tau}^{i+1}\right) p_\theta\left(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1}\right)}{p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^{i+1}\right)} \\
&= \frac{p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right) p_\theta\left(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1}\right)}{p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^{i+1}\right)}.
\end{aligned} \tag{14}$$

The term $p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^{i+1}\right)$ is not directly correlated to $\boldsymbol{\tau}^i$ at the diffusion timestep i, thus can be viewed as a constant with notation Z. □

On this basis, using Taylor series expansion [15], we can sample trajectories by the modified Gaussian resampling.

Theorem A.2. *With a sufficiently large number of reverse diffusion steps, the sampling from reverse diffusion process $p_{\theta,\phi}(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1},\boldsymbol{y})$ can be approximated by a modified Gaussian resampling. That is*

$$p_{\theta,\phi}(\boldsymbol{\tau}^i|\boldsymbol{\tau}^{i+1},\boldsymbol{y}) \approx \mathcal{N}(\boldsymbol{\tau}^i; \mu_\theta + \Sigma\nabla_{\boldsymbol{\tau}} \log p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right), \Sigma), \tag{15}$$

where $\mu_\theta = \mu_\theta(\boldsymbol{\tau}^i)$ and Σ are the mean and variance of unconditional reverse diffusion process $p_\theta(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1})$.

Proof. With the above definition, we can rewrite the transfer probability of the unconditional denoising process as

$$p_\theta(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1}) = \mathcal{N}(\boldsymbol{\tau}^i; \mu_\theta, \Sigma) \tag{16}$$

$$\log p_\theta(\boldsymbol{\tau}^i \mid \boldsymbol{\tau}^{i+1}) = -\frac{1}{2}(\boldsymbol{\tau}^i - \mu_\theta)^T \Sigma^{-1}(\boldsymbol{\tau}^i - \mu_\theta) + C \tag{17}$$

With a sufficiently large number of reverse diffusion steps, we apply Taylor expansion around $\boldsymbol{\tau}^i = \mu_\theta$ as

$$\begin{aligned}
\log p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right) = &\log p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right)|_{\boldsymbol{\tau}^i=\mu_\theta} \\
&+ \left(\boldsymbol{\tau}^i - \mu_\theta\right) \nabla_{\boldsymbol{\tau}^i} \log p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right)\Big|_{\boldsymbol{\tau}^i=\mu_\theta}.
\end{aligned}$$

Therefore, using Eq. 14, we derive

$$\log p_{\theta,\phi}(\boldsymbol{\tau}^i|\boldsymbol{\tau}^{i+1},\boldsymbol{y}) = \log p_\theta(\boldsymbol{\tau}^i|\boldsymbol{\tau}^{i+1}) + \log p_\phi(\boldsymbol{y}|\boldsymbol{\tau}^i) + C_1$$

$$\begin{aligned}
RHS = &-\frac{1}{2}\left(\boldsymbol{\tau}^i - \mu_\theta\right)^T \Sigma^{-1}\left(\boldsymbol{\tau}^i - \mu_\theta\right) \\
&+ \left(\boldsymbol{\tau}^i - \mu_\theta\right) \nabla \log p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right) + C_2
\end{aligned} \tag{18}$$

$$\begin{aligned}
RHS = &-\frac{1}{2}\left(\boldsymbol{\tau}^i - \mu_\theta - \Sigma\nabla \log p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right)\right)^T \times \Sigma^{-1} \\
&\times \left(\boldsymbol{\tau}^i - \mu_\theta - \Sigma\nabla \log p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right)\right) + C_3,
\end{aligned}$$

which means,

$$p_{\theta,\phi}(\boldsymbol{\tau}^i|\boldsymbol{\tau}^{i+1},\boldsymbol{y}) \approx \mathcal{N}(\boldsymbol{\tau}^i; \mu_\theta + \Sigma\nabla_{\boldsymbol{\tau}} \log p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right), \Sigma)$$

□

A.2. Classifier-free Diffusion Model

While classifier guidance successfully achieves conditional guidance during trajectory generation, it is nonetheless reliant on gradients from a separate trained classifier which is hard to obtain in many cases. Classifier-free guidance [18] seeks to eliminate the classifier, which achieves the same effect as classifier guidance, but without such gradients.

First of all, we define the score function of the unconditional diffusion model as

$$\epsilon_\theta(\boldsymbol{\tau}^i) = -\Sigma \nabla_{\boldsymbol{\tau}} \log p_\phi\left(\boldsymbol{\tau}^i\right). \tag{19}$$

Then, through Eq. 13 and 15, the score function of the classifier-guided diffusion model can be expressed as

$$\epsilon_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) = \epsilon_\theta(\boldsymbol{\tau}^i) - \alpha \Sigma \nabla_{\boldsymbol{\tau}} \log p_\phi\left(\boldsymbol{y} \mid \boldsymbol{\tau}^i\right), \tag{20}$$

where α is a scale hyper-parameter.

Theorem A.3. *Classifier-free guided diffusion model performs sampling with the linear combination of the conditional and unconditional score estimates as,*

$$\hat{\epsilon}_\theta = \hat{\epsilon}_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) = (1-\omega)\epsilon_\theta(\boldsymbol{\tau}^i) + \omega \epsilon_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}), \tag{21}$$

implicitly embedding guidance into the score function, with ω *the scale hyper-parameter.*

Proof. Considering there is an implicit classifier denoted as $\tilde{p}_\phi(\boldsymbol{y} \mid \boldsymbol{\tau}^i)$, with Bayes Rule [43], we can expand it as

$$\tilde{p}_\phi(\boldsymbol{y} \mid \boldsymbol{\tau}^i) \propto \tilde{p}_{\theta,\phi}(\boldsymbol{\tau}^i, \boldsymbol{y}) / p_\theta(\boldsymbol{\tau}^i).$$

Then gradient of this implicit classifier would be

$$\nabla_{\boldsymbol{\tau}} \log \tilde{p}_\phi(\boldsymbol{y} \mid \boldsymbol{\tau}^i) = \nabla_{\boldsymbol{\tau}} \log \tilde{p}_{\theta,\phi}(\boldsymbol{\tau}^i, \boldsymbol{y}) - \nabla_{\boldsymbol{\tau}} \log p_\theta(\boldsymbol{\tau}^i). \tag{22}$$

Substitute Eq. 19 in RHS, we get

$$\begin{aligned} \alpha \Sigma \nabla_{\boldsymbol{\tau}} \log \tilde{p}_\phi(\boldsymbol{y} \mid \boldsymbol{\tau}^i) &= \alpha \Sigma \nabla_{\boldsymbol{\tau}} \log \tilde{p}_{\theta,\phi}(\boldsymbol{\tau}^i, \boldsymbol{y}) \\ &\quad - \alpha \Sigma \nabla_{\boldsymbol{\tau}} \log p_\theta(\boldsymbol{\tau}^i) \\ &= -\alpha \hat{\epsilon}_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) + \alpha \epsilon_\theta(\boldsymbol{\tau}^i) \end{aligned}$$

And then substitute Eq. 20 in LHS,

$$\begin{aligned} \epsilon_\theta(\boldsymbol{\tau}^i) - \epsilon_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) &= -\alpha \hat{\epsilon}_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) + \alpha \epsilon_\theta(\boldsymbol{\tau}^i) \\ \alpha \hat{\epsilon}_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) &= \epsilon_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) + (\alpha - 1)\epsilon_\theta(\boldsymbol{\tau}^i) \\ \hat{\epsilon}_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) &= (1/\alpha)\epsilon_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) + (1 - 1/\alpha)\epsilon_\theta(\boldsymbol{\tau}^i) \end{aligned} \tag{23}$$

Let $\omega = 1/\alpha$, we obtain,

$$\hat{\epsilon}_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}) = (1-\omega)\epsilon_\theta(\boldsymbol{\tau}^i) + \omega \epsilon_\theta(\boldsymbol{\tau}^i, \boldsymbol{y}),$$

which is equal to Eq. 21. □

Therefore, in classifier-free diffusion guidance, we only need to train a single neural network to parameterize both conditional score estimator $\epsilon_\theta(\boldsymbol{\tau}^i, \boldsymbol{y})$ and unconditional score estimator $\epsilon_\theta(\boldsymbol{\tau}^i)$, where for the unconditional model we can set an empty set $\varnothing$ for the condition identifier $\boldsymbol{y}$ when predicting the score, *i.e.* $\epsilon_\theta(\boldsymbol{\tau}^i) = \epsilon_\theta(\boldsymbol{\tau}^i, \boldsymbol{y} = \varnothing)$. Following the settings of [18], we jointly train the unconditional and conditional models simply by randomly setting $\boldsymbol{y}$ to the unconditional class identifier $\varnothing$ with probability β, which balances off the diversity and the relevance of the conditional label of generated samples.

B. Pseudo-code of Training SkillDiffuser

As illustrated in Sec. 4.4, we provide the pseudocode for our SkillDiffuser's training process in Algorithm 1, detailing its sequential stages and core mechanics. Additionally, Algorithm 2 describes the inference process, illustrating its steps of skill abstraction and trajectory generation.

Algorithm 1 Training process of SkillDiffuser

Input: Dataset $\mathcal{D}$ of partially observed trajectories with paired language $\left\{\boldsymbol{\tau}_\xi = (l, \{\boldsymbol{i}_t, \boldsymbol{a}_t\}_{t=0}^{T-1})\right\}_{\xi=1}^{N}$, size of the skill set K and horizon H, pre-trained language and visual encoder Φ_{lang}, Φ_{im}

Initialize skill predictor f, conditional diffusion model $\mathcal{M}$, skill embedding model Λ and inverse dynamics model Ψ
Vector Quantization op $\mathbf{q}(\cdot)$
while *not converged* **do**
  Sample $\tau = (l, \{\boldsymbol{i}_t, \boldsymbol{a}_t\}_{t=0}^{T-1})$
  Initialize partially observed states $S = \{\Phi(\boldsymbol{i}_0)\}$
  for $k = 0 ... \lfloor \frac{T}{H} \rfloor$ **do** ▷ Sample a skill every H steps
    $z \leftarrow \mathbf{q}(f(\Phi_{lang}(l), S))$
    $\mathcal{L}_{diff} \leftarrow \mathcal{M}_{diff}(S, \Lambda(z))$ ▷ Diffusing process
    for *step* $t = 1 ... H$ **do**
      $S \leftarrow S \cup \{\Phi(\boldsymbol{i}_{kH+t+1})\}$
      $\tilde{\boldsymbol{a}}_{kH+t} \leftarrow \Psi([\boldsymbol{s}_{kH+t}, \boldsymbol{s}_{kH+t+1}], \boldsymbol{i}_{kH+t})$ ▷ Predict action using inverse dynamics model
      $\mathcal{L}_{inv} = \mathbb{E}\left[\|\boldsymbol{a}_{kH+t} - \tilde{\boldsymbol{a}}_{kH+t}\|_2^2\right]$
      Train Ψ with objective $\mathcal{L}_{inv}$
    end for
    Train f, Λ and $\mathcal{M}$ with objective $\mathcal{L}_{\text{VQ}} + \lambda \mathcal{L}_{diff}$
  end for
end while

Algorithm 2 Inference process of SkillDiffuser

Input: Initial partial observation $\boldsymbol{i}_0$ and the language instruction l, pre-trained language and visual encoder Φ_{lang}, Φ_{im}

Input: Trained skill predictor f, conditional diffusion model $\mathcal{M}$, skill embedding model Λ and inverse dynamics model Ψ

Initialize partially observed states $S = \{\Phi(\boldsymbol{i}_0)\}$
for $k = 0 ... \lfloor \frac{T}{H} \rfloor$ **do** ▷ Sample a skill every H steps
  $z \leftarrow \mathbf{q}(f(\Phi_{lang}(l), S))$
  $S' \leftarrow \mathcal{M}_{denoise}(S, \Lambda(z))$ ▷ Denoising process
  for *step* $t = 1 ... H$ **do**
    $\boldsymbol{a}_{kH+t} \leftarrow \Psi([\boldsymbol{s}_{kH+t}, \boldsymbol{s}'_{kH+t+1}], \boldsymbol{i}_{kH+t})$
    $\tilde{\boldsymbol{s}}_{kH+t+1} \leftarrow$ Env.step$(\boldsymbol{a}_{kH+t})$ ▷ Take action
    $S \leftarrow S \cup \{\tilde{\boldsymbol{s}}_{kH+t+1}\}$
  end for
end for

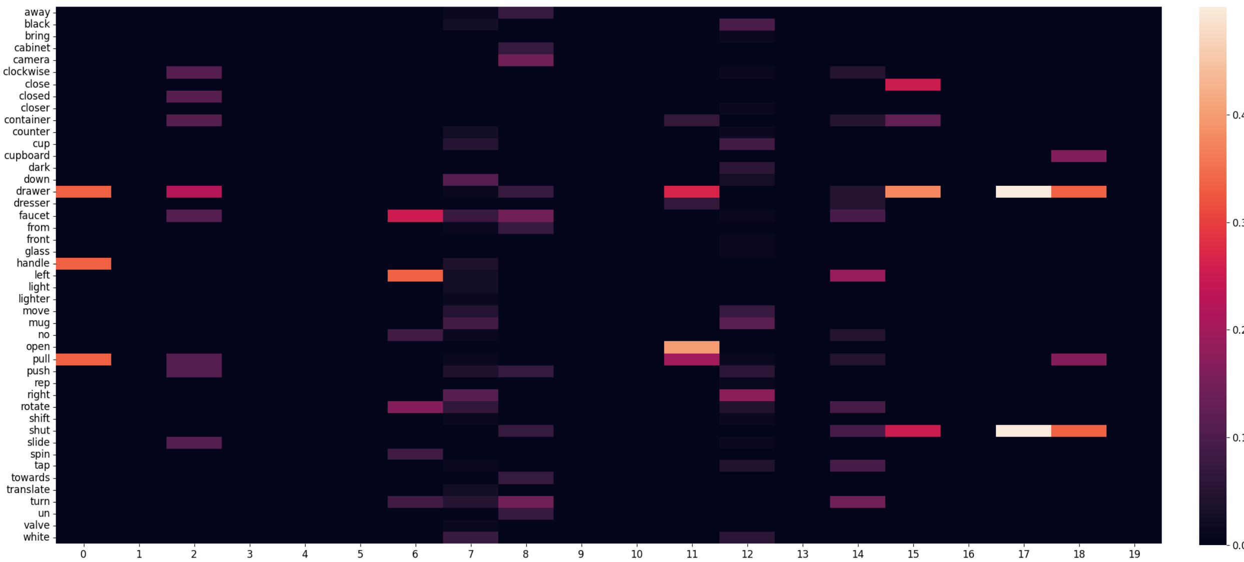

Figure 5. **Visualization of skill heat map on LOReL.** We display the word frequency associated with a skill set of size 20 in LOReL, normalized by column. The data's sparsity and distinct highlights indicate certain language tokens are uniquely linked to specific skills. There are eleven skills learned by our method.

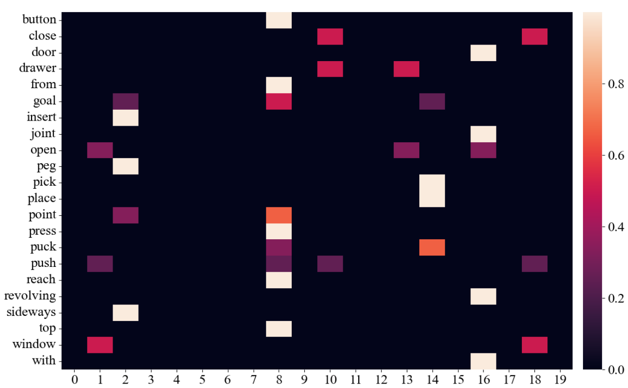

Figure 6. **Visualization of skill heat map on Meta-World Multi-Task 10 (MT10).** There are eight skills learned by our method. (zoom in for best view)

C. More Visualizations

C.1. Visualization Results of Learned Skill Set

As mentioned before, we show the visualization results of skill set on LOReL Sawyer Dataset in Fig. 5 and Meta-World Multi-Task 10 (MT10) in Fig. 6. The visualization results show that out of a 20-size skill-set, our SkillDiffuser learned 11 skills for LOReL (*e.g. pull drawer handle* [skill 0], *shut close container drawer* [skill 15], *etc.*) and 8 skills for Meta-World MT10 (*e.g. open push window* [skill 0], *open door with revolving joint* [skill 16], *etc.*). The results demonstrate strong skill abstraction abilities. For example, the skill "shut close container drawer" abstracts different expressions like "shut drawer", "shut container" into one skill semantic. In the heatmap, the presence of distinct bright spots across eleven columns strongly reaffirms the model's capability to discern and pinpoint specific skills from visual inputs, in the absence of a pre-defined skill library. This observation is not just a testament to the model's enhanced interpretative prowess over conventional diffusion-based planning approaches but also marks a remarkable stride in abstracting high-level skills into representations that are intuitively understandable by humans. Such evidence further validates the model's proficiency in sophisticated skill identification and representation.

C.2. Word Cloud of Learned Skills

We further show the word cloud of 8 learned skills of LOReL Sawyer Dataset in Figure 7. From the results, we can find that the model has successfully mastered eight key skills, each closely linked to specific tasks. These skills demonstrate strong robustness to ambiguous language instructions. For instance, skill 4 effectively abstracts the skill of "open a drawer" from ambiguous expressions such as "open a container", "pull a dresser", "pull a drawer" and random combinations of these words. Similarly, skill 6 extracts the skill of "turn a faucet to the left". This analysis indicates our method's resilience to varied and poorly defined language inputs, confirming our SkillDiffuser can competently interpret and act upon a wide range of linguistic instructions, even those that are ambiguous or incomplete. These findings provide new perspectives and methodological guidance for future research in similar fields, especially in handling complex tasks with ambiguous language instructions. We also provide the word cloud of learned skills from Meta-World MT10 dataset in Fig. 8.

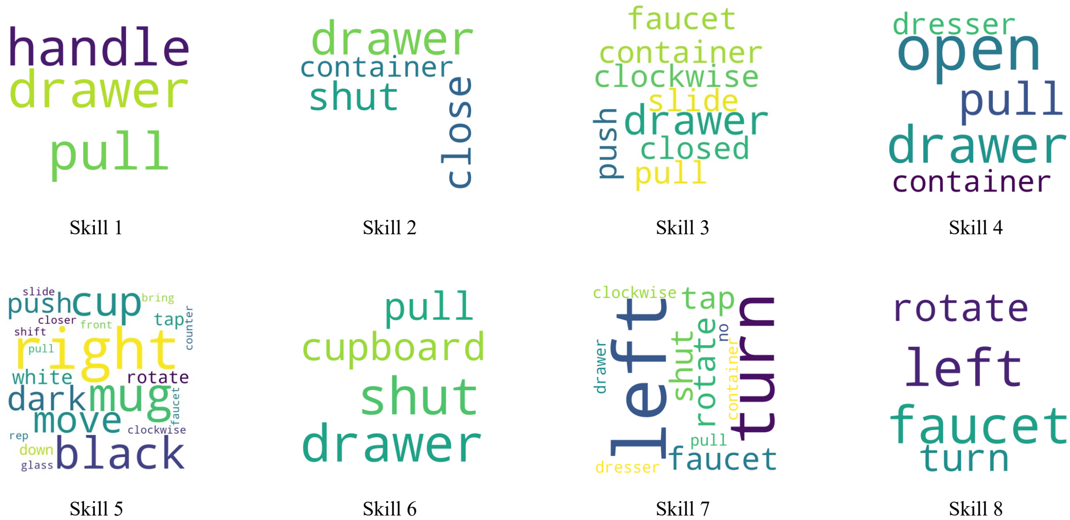

Figure 7. **Word cloud of learned skills in LOReL Sawyer Dataset.** We show eight of them here with the size corresponding to the word frequency in one skill.

Figure 8. **Word cloud of learned skills in Meta-World MT10 Dataset.** We show eight of them here with the size corresponding to the word frequency in one skill.

D. Dataset Descriptions

D.1. LOReL Sawyer Dataset

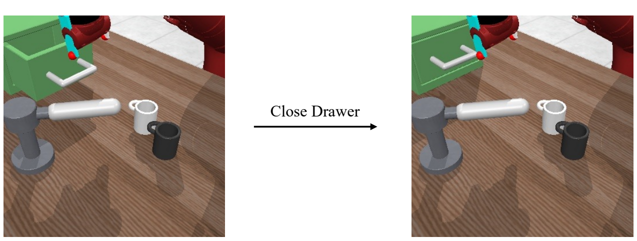

Figure 9. **A sample instance of LOReL Sawyer Dataset.** The start and goal images correspond to the instruction "close drawer".

Language-conditioned Offline Reward Learning dataset, abbreviated as LOReL [29], contains trajectories originating from a reinforcement learning buffer which is generated by a random policy. The trajectories are sub-optimal and have language annotations through crowd-sourcing. Overall, the dataset encompasses approximately 50,000 language-annotated trajectories, each within a simulated environment featuring a Sawyer robot arm, with every demonstration extending over 20 discrete steps. A typical LOReL Sawyer environment is shown in Fig. 9. We assess our approach using the same set of instructions as those outlined in the original paper [29] which are described with their objectives in Tab. 6. These evaluation tasks are along with var-

Task	Description
Closing the Drawer	Involves the robot's precise manipulation of a drawer to close it, testing spatial dynamics understanding and fine motor control.
Opening the Drawer	Requires the robot to open a drawer, emphasizing its capability in tasks that necessitate pulling and spatial navigation.
Turning the Faucet Left	Assesses the robot's precision in rotational movements for turning a faucet to the left, a nuanced everyday action.
Turning the Faucet Right	Tests the robot's adaptability in mirrored instructions, involving turning the faucet right, similar to the left turning task but in the opposite direction.
Pushing the Black Mug Right	Requires the robot to push a specific object (black mug) to the right, testing its skills in object recognition and directional movement.
Pushing the White Mug Down	Involves pushing a different object (white mug) downward, further evaluating the robot's ability to differentiate objects and execute varied motion commands.

Table 6. **Overview of tasks in LOReL Sawyer Dataset.**

Instructions
open drawer and move black mug right
pull the handle and move black mug down
move white mug right
move black mug down
close drawer and turn faucet right
close drawer and turn faucet left
turn faucet left and move white mug down
turn faucet right and close drawer
move white mug down and turn faucet left
close the drawer, turn the faucet left and move black mug right
open drawer and turn faucet counterclockwise
slide the drawer closed and then shift white mug down

Table 7. **LOReL composition tasks**

Task Identifier	Language Instruction
window-close	push and close a window
window-open	push and open a window
door-open	open a door with a revolving joint
peg-insert-side	insert a peg sideways to the goal point
drawer-open	open a drawer
pick-place	pick a puck, and place the puck to the goal
reach	reach the goal point
button-press-topdown	press the button from the top
push	push the puck to the goal point
drawer-close	push and close a drawer

Table 8. **Annotated instructions for Meta-World MT10 tasks.**

ious rephrases of instructions which modify either the noun ("unseen noun"), the verb ("unseen verb"), both ("unseen noun+verb"), or entail a complete rewrite of the task ("human provided"), leading to a total of 77 distinct instructions for all six tasks. This structure of tasks and rephrases enables a comprehensive assessment of the robot's ability to interpret and execute a wide range of language-based commands within the simulated environment.

D.2. LOReL Composition Tasks

We follow the same settings as LISA [13] to create 12 new composition tasks through combining original evaluation instructions as shown in Tab. 7.

Additionally, we also incorporate tasks such as "move white mug right" and "move black mug down" to explore the composition of skills related to colors (e.g., black and white) and directions (e.g., right and down). This aims to explore whether such skills can be combined to fulfill complex instructions.

D.3. Meta-World Dataset

The Meta-World dataset establishes a new benchmark in the field of multi-task and meta-reinforcement learning, offering 50 unique robotic manipulation tasks. These tasks range from simple to complex operations, providing researchers with a diverse testing ground. Each task is meticulously designed to ensure both challenge and common structural features that can be leveraged in multi-task and meta-learning algorithms. This design makes Meta-World an ideal choice for assessing the effectiveness and adaptability of algorithms in complex and variable task environments.

Particularly, the Multi-Task 10 (MT10) subset comprises 10 carefully selected tasks, where algorithms are trained and subsequently tested on the same set of tasks. As shown in Fig. 13, MT10 challenges algorithms' learning and generalization capabilities in a multi-task environment, with the aim to evaluate the consistency and efficiency of algorithms in mastering multiple tasks, as well as their adaptability and robustness in the face of diverse tasks. As there is currently no widely-recognized instruction labeling of MT10, we provide our annotations here in Tab. 8.

We sample 100 trajectories for each task of MT10 and form the expert dataset of 1000 trajectories. We have released our dataset with image observations on `https://skilldiffuser.github.io`.

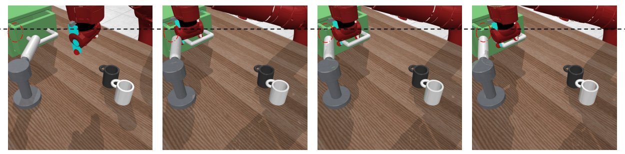

Figure 10. **Resulting images from applying skill 11 of Fig. 5.** The black dashed line is a horizontal reference and please pay attention to the red oval region. (zoom in for best view)

E. More Ablations

E.1. Ablation Study on Skill Interpretability

Resulting Images from Applying Discrete Skills We visualize resulting images from applying skill 11 of Fig. 5 which has grounding of "open, drawer, pull, dresser, container" (ranked from high frequency to low ones), consistent with its actual actions in Fig. 10. We can clearly observe a behavior of pulling the drawer. And we would like to clarify not all skills have clear semantic or action correspondences, while some do.

E.2. Ablation Study on Condition Guidance Weight

Classifier-free guidance is widely used in generative model domain for its ability to act as temperature control when setting guidance weight above 1 during inference. In all of our experiments, we set guidance weight to 1.2 by default. But we also conduct ablation study on the condition guidance weight here in Tab. 9. From the results, we find the guidance weight slightly greater than 1 helps the planner's performance, while excessive weight hurts.

Guidance Weight	**1.0**	**1.2**	**1.8**	**3.0**	**5.0**
Success Rate on Seen Tasks	39.33%	46.67%	38.86%	39.03%	33.50%

Table 9. Ablation on guidance weight. (5 episodes over 3 seeds.)

F. More Results

F.1. Task-wise Performance on LOReL Dataset

We further demonstrate the performance of our method and other baselines on LOReL Sawyer dataset in Fig 11 and 12. As can be seen from the figures, especially from Fig. 12, our method's average performance on 5 rephrases is nearly 10 percentage points higher than the previous SOTA, which demonstrates its strong robustness against ambiguous language instructions.

F.2. Task-wise Performance on Meta-World

We also provide the task-wise success rates on Meta-World MT10 dataset in Fig. 14, achieved by Flat R3M [30], Language-conditioned Diffuser and SkillDiffuser. The average performance is shown separately in the right figure. From our experimental outcomes, it is clear to observe that our SkillDiffuser demonstrates commendable performance, particularly excelling in tasks involving mirrored instructions. SkillDiffuser exhibits an average performance enhancement of over 5% than previous language-conditioned Diffuser, which highlights the model's advanced capability in understanding complex and ambiguous instructions compared to traditional methods. It showcases SkillDiffuser's superior use of hierarchical architecture that employs interpretable skill learning for diffusion-based planners to better generate future trajectories.

G. Implementation Details

G.1. Hyper-parameters

Generally, we follow the settings illustrated in [13] with details specified in the following Tab. 10.

Hyper-parameter	LOReL	Meta-World
Skill Predictor Transformer Layers	1	1
Skill Predictor Embedding Dim	128	128
Skill Predictor Transformer Heads	4	4
Skill Set Code Dim	16	16
Skill Set Size	20	20
Dropout	0.1	0.1
Batch Size	256	64
Skill Predictor Learning Rate	1e-6	1e-5
Conditional Diffuser Learning Rate	1e-3	5e-3
Condition Guidance Weight	1.2	1.2
Inverse Dynamics Model Learning Rate	1e-3	5e-4
Diffuser Loss Weight	0.005	0.01
Horizon	8	8
VQ EMA Update	0.99	0.99
Skill Predictor and Diffuser Optimizer	Adam	Adam
Inverse Dynamics Model Optimizer	Adam	Adam

Table 10. **Hyper-parameters of SkillDiffuser.**

G.2. Architecture Details

1. We use 1 layer Transformer network for the skill predictor and follow the implementation of VQ-VAE [45] to achieve VQ operation.
2. The size of skill set is set to 20 and the planning horizon is set to 8 for all implementations.
3. A temporal U-Net [36] with 6 repeated residual blocks is employed to model the noise ϵ_θ of the diffusion process. Each block is comprised of two temporal convolutions, each followed by group norm [47], and a final Mish non-linearity [26]. Timestep and skill embeddings are generated by two separate single fully-connected layer and added to the activation output after the first temporal convolution of each block.

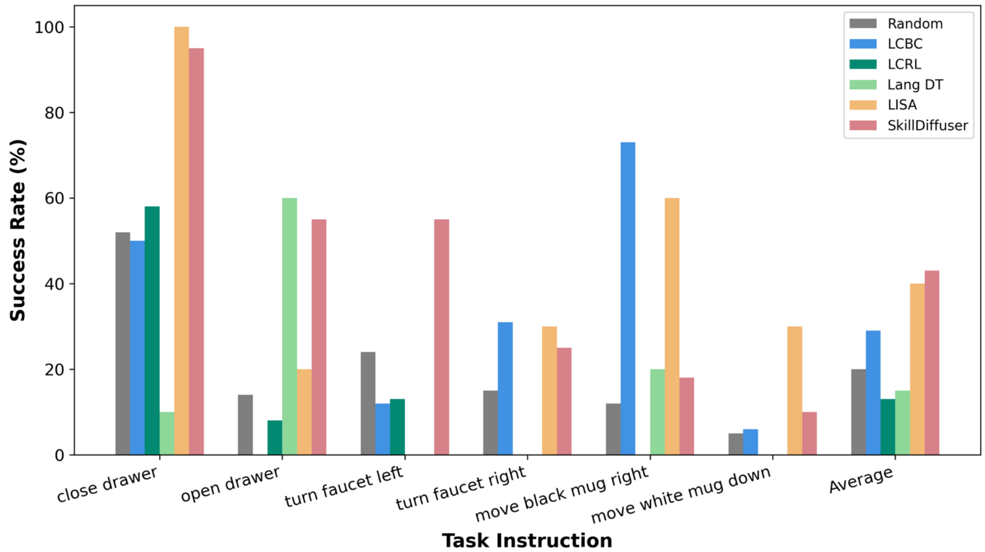

Figure 11. **Task-wise success rates (in %) on LOReL Sawyer Dataset.**

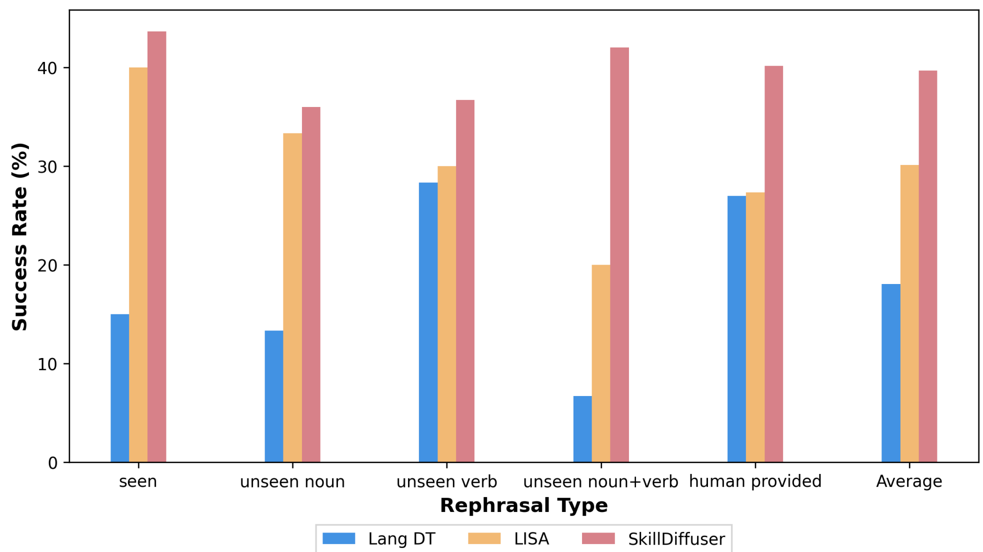

Figure 12. **Rephrasal-wise success rates (in %) on LOReL Sawyer Dataset.**

G.3. Training Details

1. We train our model with one NVIDIA A100 Core Tensor GPU for about 45 hours in LOReL Sawyer dataset and about 24 hours in Meta-World MT10 dataset (1000 trajectories in total).
2. In both LOReL and Meta-World dataset, the skill predictor and diffusion model are trained with Adam optimizer [21] using a learning rate of 1×10^{-3} for the diffusion model, 1×10^{-6} for the LOReL skill predictor while 1×10^{-5} for Meta-World skill predictor. We only update parameters of Meta-World skill predictor every ten iterations. The inverse dynamics model is updated with Adam optimizer as well.
3. The batch size is set to 256 for LOReL Sawyer dataset and 64 for Meta-World MT10 dataset.
4. The training steps of the diffusion model are $5K$ for LOReL Sawyer dataset and $8K$ for Meta-World MT10 dataset. And the training epochs of the skill predictor are 500 for both datasets.
5. The planning horizon T of diffusion model is set to 100 and the denoising steps are set to 200 for all tasks.

References

[1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i

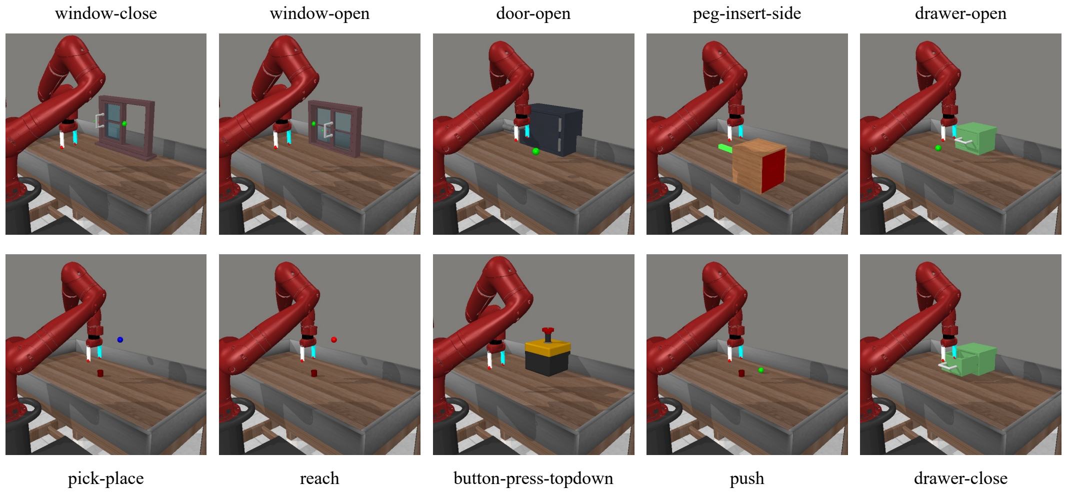

Figure 13. **Partially visual observations of all the 10 tasks in Meta-World MT10 Dataset.**

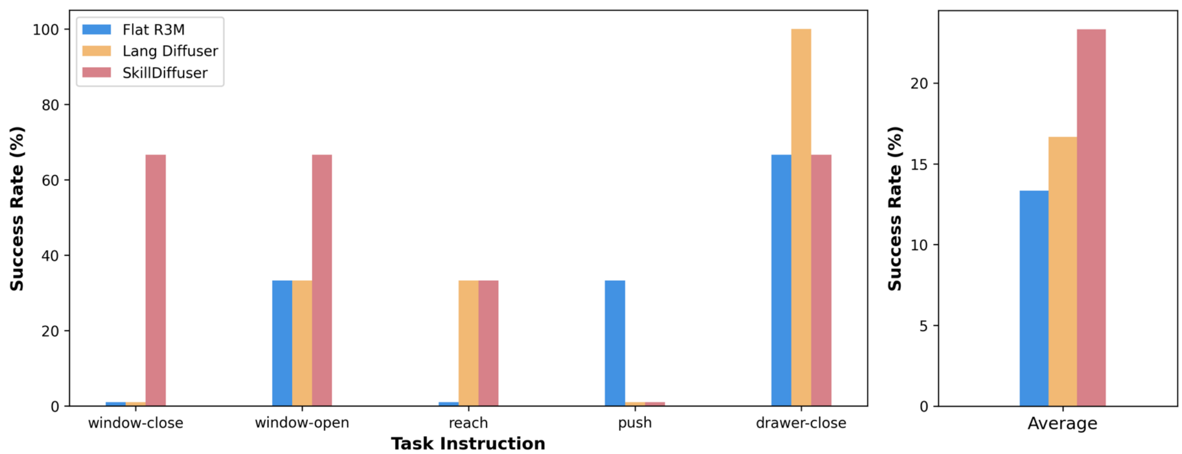

Figure 14. **Task-wise success rates (in %) on Meta-World MT10 Dataset.**

can, not as i say: Grounding language in robotic affordances. 2022. 1

[2] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B Tenenbaum, Tommi S Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2022. 1, 3, 5, 6

[3] Joao Carvalho, An T Le, Mark Baierl, Dorothea Koert, and Jan Peters. Motion planning diffusion: Learning and planning of robot motions with diffusion models. *arXiv preprint arXiv:2308.01557*, 2023. 3

[4] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021. 7

[5] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023. 1, 3

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 1, 3

[7] Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 2023. 1, 3, 6

[8] Yan Duan, Marcin Andrychowicz, Bradly Stadie, OpenAI Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. *Advances in neural information processing systems*, 30, 2017. 2

[9] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018. 2

[10] William Feller. On the theory of stochastic processes, with

particular reference to applications. In *Selected Papers I*, pages 769–798. Springer, 2015. 3

[11] Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, pages 357–368. PMLR, 2017. 2

[12] Divyansh Garg, Shuvam Chakraborty, Chris Cundy, Jiaming Song, and Stefano Ermon. Iq-learn: Inverse soft-q learning for imitation. *Advances in Neural Information Processing Systems*, 34:4028–4039, 2021. 2

[13] Divyansh Garg, Skanda Vaidyanath, Kuno Kim, Jiaming Song, and Stefano Ermon. Lisa: Learning interpretable skill abstractions from language. *Advances in Neural Information Processing Systems*, 35:21711–21724, 2022. 2, 3, 6, 7, 8, 5

[14] Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. *Advances in neural information processing systems*, 2023. 1, 2

[15] Bernd Heidergott, editor. *Taylor Series Expansions*, pages 179–263. Springer US, Boston, MA, 2007. 1

[16] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016. 2

[17] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. 3

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 1, 3, 2

[19] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022. 1, 3

[20] Yiding Jiang, Shixiang Shane Gu, Kevin P Murphy, and Chelsea Finn. Language as an abstraction for hierarchical deep reinforcement learning. *Advances in Neural Information Processing Systems*, 32, 2019. 7

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 7

[22] Alexander C Li, Carlos Florensa, Ignasi Clavera, and Pieter Abbeel. Sub-policy adaptation for hierarchical reinforcement learning. *arXiv preprint arXiv:1906.05862*, 2019. 3

[23] Zhixuan Liang, Yao Mu, Mingyu Ding, Fei Ni, Masayoshi Tomizuka, and Ping Luo. Adaptdiffuser: Diffusion models as adaptive self-evolving planners. In *International Conference on Machine Learning*, 2023. 1, 3, 6

[24] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, pages 281–297. Oakland, CA, USA, 1967. 5

[25] Valentyn Melnychuk, Dennis Frauen, and Stefan Feuerriegel. Causal transformer for estimating counterfactual outcomes. In *International Conference on Machine Learning*, pages 15293–15329. PMLR, 2022. 7

[26] Diganta Misra. Mish: A self regularized non-monotonic activation function. In *31st British Machine Vision Conference 2020, BMVC 2020, Virtual Event, UK, September 7-10, 2020*, 2020. 6

[27] Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018. 3

[28] Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. *Advances in neural information processing systems*, 31, 2018. 2

[29] Suraj Nair, Eric Mitchell, Kevin Chen, Silvio Savarese, Chelsea Finn, et al. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Conference on Robot Learning*, pages 1303–1315. PMLR, 2022. 2, 6, 7, 4

[30] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, pages 892–909. PMLR, 2023. 2, 4, 7, 8, 6

[31] Fei Ni, Jianye Hao, Yao Mu, Yifu Yuan, Yan Zheng, Bin Wang, and Zhixuan Liang. Metadiffuser: Diffusion model as conditional planner for offline meta-rl. In *International Conference on Machine Learning*, 2023. 3

[32] Scott Niekum, Sarah Osentoski, George Konidaris, Sachin Chitta, Bhaskara Marthi, and Andrew G Barto. Learning grounded finite-state representations from unstructured demonstrations. *The International Journal of Robotics Research*, 34(2):131–157, 2015. 2

[33] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3 (1):88–97, 1991. 2

[34] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994. 3

[35] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 4

[36] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015. 6

[37] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668. JMLR Workshop and Conference Proceedings, 2010. 2

[38] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011. 2

[39] V Sanh. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *Proceedings of Thirty-third Conference on Neural Information Processing Systems*, 2019. 6

[40] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019. 2

[41] Avi Singh, Eric Jang, Alexander Irpan, Daniel Kappler, Murtaza Dalal, Sergey Levinev, Mohi Khansari, and Chelsea Finn. Scalable multi-task imitation learning with autonomous improvement. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2167–2173. IEEE, 2020. 2

[42] Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and Heni Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33: 13139–13150, 2020. 7

[43] Alan Stuart and J Keith Ord. Kendall's advanced theory of statistics. vol. 1: Distribution theory. *Kendall's advanced theory of statistics. Vol. 1: Distribution theory*, 1994. 2

[44] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4950–4957, 2018. 6

[45] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 2, 5, 6

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 5

[47] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 6

[48] Danfei Xu, Suraj Nair, Yuke Zhu, Julian Gao, Animesh Garg, Li Fei-Fei, and Silvio Savarese. Neural task programming: Learning to generalize across hierarchical tasks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3795–3802. IEEE, 2018. 2

[49] Mengdi Xu, Yikang Shen, Shun Zhang, Yuchen Lu, Ding Zhao, Joshua Tenenbaum, and Chuang Gan. Prompting decision transformer for few-shot policy generalization. In *international conference on machine learning*, pages 24631–24645. PMLR, 2022. 2, 8

[50] Long Yang, Zhixiong Huang, Fenghao Lei, Yucun Zhong, Yiming Yang, Cong Fang, Shiting Wen, Binbin Zhou, and Zhouchen Lin. Policy representation via diffusion probability model for reinforcement learning. *arXiv preprint arXiv:2305.13122*, 2023. 1

[51] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020. 2, 6

[52] Jesse Zhang, Haonan Yu, and Wei Xu. Hierarchical reinforcement learning by discovering intrinsic options. *arXiv preprint arXiv:2101.06521*, 2021. 3