SALT: Learning State- and Temporally-Abstracted World Models for Offline Long-Horizon Decision-Making

William Huang [1] **Benjamin Freed** [2]

Abstract

Long-horizon decision making requires agents to reason over temporally extended behaviors rather than individual low-level actions. Existing skill-based methods reduce the effective planning horizon but often plan in the raw environment state space, while latent world models learn compact representations but typically lack temporal abstraction. We introduce State-Abstracted Latent skills for Temporal planning (SALT), a framework that jointly learns temporal and state abstractions from offline trajectories to support planning in a compact latent space. In preliminary experiments on AntMaze-medium-diverse, SALT achieves $71.7 \pm 5.1\%$ offline success after 50 epochs without model or planner finetuning. With online finetuning, SALT improves to $78.3 \pm 4.7\%$ success and outperforms PPO and SAC trained from scratch under the same episode budget.

1. Introduction

Human decision-making is routinely made over long temporal horizons in high-dimensional environments. Consider the example of a self-driving car driving to a grocery store. Rather than reasoning over every small steering change or acceleration adjustment independently, the car must plan over temporally extended behaviors: merging onto a road, navigating an intersection, changing lanes, and turning into a parking lot. Each of these behaviors takes place over many low-level control steps but can be treated as a higher-level decision when planning over long distances. Throughout its trip, the car will also encounter scenes rich with both control-relevant information (eg. traffic lights), and control-irrelevant information (eg. clouds).

This structure suggests that effective long-horizon decision-making requires two complementary forms of abstraction: temporal abstraction, which groups low-level actions into reusable skills, and state abstraction, which compresses high-dimensional observations into control-relevant representations. The main challenge is to learn these abstractions jointly, so that planning can occur over both temporally extended behaviors and compact abstract states.

Temporal abstraction has been widely explored in deep reinforcement learning (RL) to solve long-horizon tasks. For these tasks, effective sequential decision-making typically emerges from temporally extended behaviors. This observation motivates the use of temporal abstraction, where agents plan over skills or options that span multiple low-level timesteps (Achiam et al., 2018; Ajay et al., 2021; Eysenbach et al., 2018; Freed et al., 2023; Pertsch et al., 2021; Sharma et al., 2020b; Sutton et al., 1999). Learning these abstractions from datasets could allow agents to plan more efficiently by searching over abstract skill sequences rather than primitive actions in the original observation space.

A complementary angle focuses on *state abstraction* via learned world models. Rather than reasoning directly in high-dimensional observation spaces, model-based approaches learn compact latent representations that capture the essential structure for prediction and control (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2020; Hansen et al., 2022). Dreamer-style models, for instance, learn recurrent latent dynamics that enable imagination entirely in latent space (Hafner et al., 2023). While such methods provide powerful abstractions for control, they typically rely on single-step latent transitions, limiting their ability to capture temporally extended behavior.

Recent work has begun to combine temporal and state abstraction. Director, for example, develops a hierarchical framework in which a high-level policy proposes latent goals while a low-level policy acts to realize them (Hafner et al., 2022). Other work has studied how compact state representations and temporally extended actions can interact to support decision-making (Zadem et al., 2024). However, because these approaches are primarily online, they require

[1]Department of Computer Science, Yale University, New Haven, USA [2]Robotics Institute, Carnegie Mellon University, Pittsburgh, USA. Correspondence to: William Huang <william.huang@yale.edu>.

Accepted at ICML 2026 Workshop on Decision-Making from Offline Datasets to Online Adaptation: Black-Box Optimization to Reinforcement Learning.

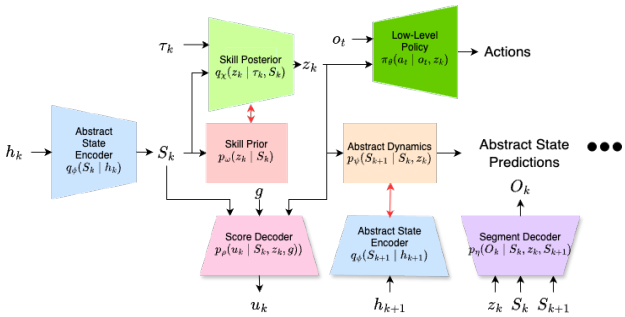

Figure 1. **Temporally and State-Abstracted Model Architecture.** SALT infers abstract boundary states and latent skills from offline trajectory segments, predicts skill-conditioned transitions in the learned abstract state space, and scores imagined transitions for goal-conditioned planning.

environment interaction to learn useful abstractions. This can be costly in sparse-reward long-horizon domains.

Offline datasets offer a natural setting for learning such abstractions before additional interaction is available. Prior work has learned reusable skills from offline trajectories (Ajay et al., 2021; Freed et al., 2023; Pertsch et al., 2021) or latent dynamics for model-based control (Hafner et al., 2019; 2020). However, these lines of work typically emphasize either temporal abstraction or state abstraction in isolation, rather than integrating both into a single skill-conditioned abstract world model.

A natural way to learn such a model is through variational inference, in which skills and states are treated as latent variables that can be inferred from observed trajectories. This perspective has been used separately for temporal abstraction, as in OPOSM's ELBO-based skill learning from offline trajectory segments (Freed et al., 2023), and for state abstraction, as in PlaNet and Dreamer-style latent state-space models (Hafner et al., 2019; 2020; 2023). This leaves the following question unanswered: how can we jointly infer latent skills and compact abstract states in a single model that predicts skill-level transitions for long-horizon planning?

2. Model-Based Temporal and State Abstraction

We introduce *State-Abstracted Latent skills for Temporal planning* (SALT). SALT is a framework that learns temporally extended skills and a temporally and state-abstracted world model (TSAWM) from offline trajectories. Together, these components enable planning over skill transitions in a learned abstract state space. Thus, SALT reduces the planning horizon while preserving low-level executability.

Our main contribution is a unified offline framework that jointly learns 1) latent abstract boundary states for state abstraction, 2) temporally extended latent skills to reduce the planning horizon, and 3) skill-conditioned abstract dynamics with goal-conditioned scoring for planning. SALT differentiates itself from OPOSM by planning in a learned abstract state space instead of the original environment state space. Unlike Dreamer-style models, SALT predicts multi-step skill transitions rather than single-step latent transitions.

2.1. Learning a State- and Temporally Abstracted World Model from Offline Data

Let an observed low-level trajectory be $o_{0:T}$, $a_{0:T-1}$, where o_t is the observation and a_t is the action. We divide the trajectory into K segments of length H, so that $T = KH$. Each segment is represented by an abstract boundary state S_k and a latent skill z_k. A skill is a temporally extended behavior primitive represented by a latent variable z_k that conditions a low-level policy $\pi_\theta(a_t \mid o_t, z_k)$ for a fixed horizon H. The abstract state S_k summarizes the trajectory prefix up to boundary k, while z_k summarizes the behavior executed over the next H low-level time steps.

For segment k, define $O_k = o_{kH+1:(k+1)H}$, $A_k = a_{kH:(k+1)H-1}$, $\tau_k = (O_k, A_k)$. We propose to use a filtering posterior for abstract states and a segment-level posterior for skills:

$$q(S_{0:K}, z_{0:K-1} \mid o_{0:T}, a_{0:T-1}) = \prod_{k=0}^{K} q_\phi(S_k \mid h_k) \prod_{k=0}^{K-1} q_\chi(z_k \mid \tau_k, S_k), \tag{1}$$

where the history available at abstract boundary k is $h_k = (o_{0:kH}, a_{0:kH-1})$.

The generative model factorizes across abstract segments:

$$\begin{aligned} p(o, a, u, S, z \mid g) &= p(S_0)p(o_0 \mid S_0) \prod_{k=0}^{K-1} \Big[p_\omega(z_k \mid S_k) p_\psi(S_{k+1} \mid S_k, z_k) \\ &\quad \cdot p_\eta(O_k \mid S_k, z_k, S_{k+1}) p_\rho(u_k \mid S_k, z_k, g) \\ &\quad \cdot \prod_{t=kH}^{(k+1)H-1} \pi_\theta(a_t \mid o_t, z_k) \Big]. \end{aligned} \tag{2}$$

Here p_ω is a state-conditioned skill prior, p_ψ is the abstract dynamics model, p_η reconstructs the observation segment, p_ρ is a goal-conditioned score decoder, and π_θ is the skill-conditioned low-level policy. The score decoder predicts a segment-level planning signal u_k, such as negative distance-to-goal in AntMaze environments or task completion in Franka Kitchen.

Maximizing the marginal likelihood yields the evidence

lower bound:

$$\begin{aligned}\mathcal{L} =& \mathbb{E}_q \Bigg[\log p(o_0 \mid S_0) + \sum_{k=0}^{K-1} \Big(\log p_\eta(O_k \mid S_k, z_k, S_{k+1}) \\ &+ \log p_\rho(u_k \mid S_k, z_k, g) \\ &+ \sum_{t=kH}^{(k+1)H-1} \log \pi_\theta(a_t \mid o_t, z_k) \Big) \Bigg] \\ &- D_{\mathrm{KL}}\left(q_\phi(S_0 \mid h_0) \| p(S_0)\right) \\ &- \sum_{k=1}^{K} \mathbb{E}_q \left[D_{\mathrm{KL}}\left(q_\phi(S_k \mid h_k) \| p_\psi(S_k \mid S_{k-1}, z_{k-1})\right)\right] \\ &- \sum_{k=0}^{K-1} \mathbb{E}_q \left[D_{\mathrm{KL}}\left(q_\chi(z_k \mid \tau_k, S_k) \| p_\omega(z_k \mid S_k)\right)\right].\end{aligned} \tag{3}$$

The reconstruction and policy terms encourage each skill to explain the low-level behavior within a segment. The abstract-state KL regularizes predicted boundary states toward the learned abstract dynamics, while the skill KL regularizes inferred skills to a state-conditioned prior. The score-decoding term trains the model to assign a goal-conditioned utility to each abstract skill transition. Together, these terms produce a compact latent world model in which planning can occur over K scored abstract transitions instead of T low-level actions. The corresponding architecture diagram is shown in Fig. 1. Following OPOSM, we also use an EM-style training procedure to encourage skills to capture their causal influence on behavior. In the E-step, the skill encoder is updated toward an action-conditioned posterior that favors skills which both are likely under the prior and explain the observed actions:

$$p^\star(z_k \mid S_k, \tau_k) \propto p_\omega(z_k \mid S_k) \prod_{t=kH}^{(k+1)H-1} \pi_\theta(a_t \mid o_t, z_k). \tag{4}$$

In the M-step, the generative components are updated while holding the encoders fixed.

2.2. Planning with TSAWM

Once the abstract world model has been learned, the agent plans over temporally extended skills in the learned abstract state space. Given a current abstract state $\hat{S}_0$ and goal g, we use CEM to optimize a sequence of normalized skill variables $\epsilon_{0:K-1}$. Each ϵ_k is mapped to a skill through the state-conditioned prior, $z_k = \mu_\omega(\hat{S}_k) + \sigma_\omega(\hat{S}_k) \odot \epsilon_k$, $p_\omega(z_k \mid \hat{S}_k) = \mathcal{N}(\mu_\omega(\hat{S}_k), \sigma_\omega(\hat{S}_k))$. Planning in this whitened space gives a better-conditioned search problem, since likely skills correspond to values near a unit Gaussian. Candidate skill sequences are then rolled out through the abstract dynamics and scored using our trained goal-conditioned segment score model.

Table 1. AntMaze-medium-diverse success rates. SALT is evaluated after 50 epochs without model or planner fine-tuning.

Method	Success Rate
BC	0.0
LLP	31.2 ± 5.74
CQL	53.7 ± 6.1
OPOSM	$\mathbf{78.29 \pm 4.32}$
SALT (ours)	$\mathbf{71.7 \pm 5.1}$

We use model predictive control where, after CEM selects a sequence, the agent executes only the first skill through the low-level policy for H steps before replanning. This makes planning more robust to errors in the abstract model.

3. Experiments

We evaluate SALT on AntMaze-medium-diverse, a sparse-reward long-horizon navigation task from D4RL. The task requires a quadruped agent to compose many locomotion behaviors in order to reach diverse goals in a maze. Our initial experiments are designed to answer three questions: 1) whether SALT can match or improve upon offline temporal-abstraction methods, 2) whether the learned abstract model supports meaningful planning, and 3) whether the offline model provides a useful initialization for online finetuning.

3.1. Offline Long-Horizon Planning Benchmarks

SALT is trained on fixed-length trajectory windows with segment length $H = 10$, planning horizon $K = 4$, abstract state dimension 32, and skill dimension 256. At evaluation time, the agent first infers the current abstract state. Then, it plans in whitened skill space using CEM, rolls out candidate skill sequences through the learned abstract dynamics, executes the first selected skill using the low-level policy, and then replans.

We compare SALT to various offline methods: behavior cloning (BC), which directly imitates dataset actions without planning; low-level planning (LLP), which performs model-based planning over primitive actions or low-level states; CQL, a strong value-based offline RL baseline; and OPOSM, a temporally abstract offline planning method.

Table 1 reports the main offline comparison, with additional baselines provided in Appendix 3. SALT achieves 71.7 ± 5.1 success after only 50 epochs of training, without fine-tuning either the world model or the CEM planner. This substantially improves over low-level planning and behavior cloning, and is competitive with OPOSM, despite planning in a learned abstract state space rather than the raw environment state space. These results suggest that state abstraction can preserve the benefits of temporally extended behavioral primitives while enabling planning in a compact

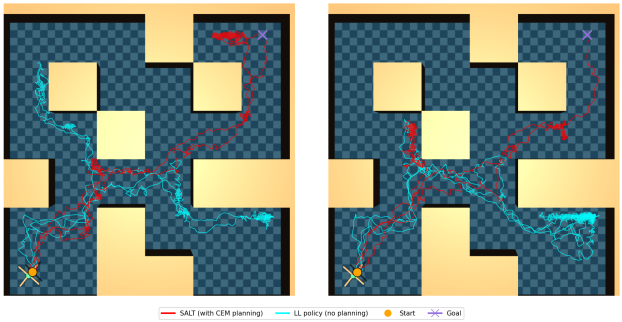

Figure 2. **Qualitative Comparison of SALT and Low-Level Policy on AntMaze-medium-diverse.** Three trajectories are shown for each method across two different sets of rollouts toward the same goal from a fixed start state. SALT with CEM planning (red) produces directed paths toward the goal by composing skills through the TSAWM, while the low-level policy without planning (blue) often wanders or fails to make consistent progress.

latent space.

3.2. Visualization of Planned Predictions

To qualitatively examine the learned abstract model, we visualize planned skill compositions in AntMaze-medium-diverse. For each goal, we execute multiple rollouts from a fixed start state and overlay the resulting trajectories. As seen in Fig. 2, SALT produces trajectories that consistently move toward the goal. In contrast, the low-level policy without planning often wanders or fails to make sustained progress toward the goal.

3.3. Online Finetuning

Finally, we study whether we can improve offline SALT with online replay. Starting from the offline checkpoint, we run an online finetuning loop. At each episode, the agent collects a rollout using CEM planning with 100 CEM iterations and adds the resulting trajectory as overlapping T-step chunks into a circular replay buffer of capacity 5,000. The model is then updated for 10 E-M gradient steps per episode on batches drawn from a combination of 80% offline data and 20% online buffer. Importantly, the segment score model is frozen during online finetuning to prevent online failures from corrupting the learned reward signal. We compare online-finetuned SALT against PPO and SAC trained online from scratch under the same episode budget of 5,000 episodes. For AntMaze, we evaluate success over 300 trials as the fraction reaching within 1 unit of the goal.

Online finetuning improves SALT over the offline baseline after only 500 episodes of online interaction, and outperforms PPO and SAC which lack access to the offline dataset and are trained from scratch under the same episode budget, as shown in Table 2.

Table 2. Success rate on AntMaze-medium-diverse comparing online methods.

Method	Success Rate
PPO (online)	0.0%
SAC (online)	0.0%
SALT (online, ours)	$\mathbf{78.3 \pm 4.7}\%$

4. Conclusion

We introduced SALT, a framework for learning state- and temporally abstracted world models from offline trajectories. By combining latent skills, abstract boundary states, and skill-conditioned latent dynamics, SALT enables planning over temporally extended transitions in a compact state space. Preliminary results on AntMaze-medium-diverse show that SALT achieves competitive offline performance after limited training and can be further improved through online finetuning. Future work will scale this evaluation to additional long-horizon domains, including AntMaze-large, Maze2D, and Franka Kitchen. We also aim to test an information-theoretic training objective for improved stability by replacing the abstract-state KL with a structural constraint on the information retained in the latent states.

References

Achiam, J., Edwards, H., Amodei, D., and Abbeel, P. Variational option discovery algorithms, 2018. URL https://arxiv.org/abs/1807.10299.

Ajay, A., Kumar, A., Agrawal, P., Levine, S., and Nachum, O. Opal: Offline primitive discovery for accelerating offline reinforcement learning. In *International Conference on Learning Representations*, 2021.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function, 2018. URL https://arxiv.org/abs/1802.06070.

Freed, B., Venkatraman, S., Sartoretti, G., Schneider, J., and Choset, H. Learning temporally abstract world models without online experimentation. In *International Conference on Machine Learning*, 2023.

Ghasemipour, S. K. S., Schuurmans, D., and Gu, S. S. EMaQ: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pp. 3682–3691. PMLR, 2021.

Ha, D. and Schmidhuber, J. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, 2019.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Hafner, D., Lee, K.-H., Fischer, I., and Abbeel, P. Deep hierarchical planning from pixels, 2022. URL https://arxiv.org/abs/2206.04114.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Hansen, N., Su, H., and Wang, X. Temporal difference learning for model predictive control. In *International Conference on Machine Learning*, 2022.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191, 2020.

Pertsch, K., Lee, Y., and Lim, J. J. Spirl: Learning skill priors for reinforcement learning. In *International Conference on Learning Representations*, 2021.

Pomerleau, D. A. ALVINN: An autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, volume 1, 1988.

Sharma, A., Ahn, M., Levine, S., Kumar, V., Hausman, K., and Gu, S. Emergent real-world robotic skills via unsupervised off-policy reinforcement learning. *arXiv preprint arXiv:2004.12974*, 2020a.

Sharma, A., Gu, S., Levine, S., Kumar, V., and Hausman, K. Dynamics-aware unsupervised discovery of skills. In *International Conference on Learning Representations*, 2020b.

Sutton, R. S., Precup, D., and Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2): 181–211, 1999.

Yu, T., Kumar, A., Rafailov, R., Rajeswaran, A., Levine, S., and Finn, C. COMBO: Conservative offline model-based policy optimization. In *Advances in Neural Information Processing Systems*, volume 34, pp. 28954–28967, 2021.

Zadem, M., Mover, S., and Nguyen, S. M. Reconciling spatial and temporal abstractions for goal representation, 2024. URL https://arxiv.org/abs/2401.09870.

Table 3. AntMaze-medium-diverse success-rate comparison. The SALT result reflects an early 50-epoch checkpoint without model or planner fine-tuning.

Task	SALT (ours)	OPOSM	LLP	CQL+OPAL	BC+OPAL	BC	BEAR	EMaQ	CQL	CQL+Off-DADS	COMBO
Medium	71.7 ± 5.1	78.29 ± 4.32	31.2 ± 5.74	81.1 ± 3.1	24.0 ± 4.8	0.0	8.0	0.0	53.7 ± 6.1	59.6 ± 2.9	17.3 ± 4.3

Note: LLP denotes low-level planning (Freed et al., 2023); OPAL denotes Offline Primitive Discovery for Accelerating Offline Reinforcement Learning (Ajay et al., 2021); BC denotes behavior cloning (Pomerleau, 1988); BEAR denotes Bootstrapping Error Accumulation Reduction (Kumar et al., 2019); EMaQ denotes Expected-Max Q-learning (Ghasemipour et al., 2021); CQL denotes Conservative Q-Learning (Kumar et al., 2020); Off-DADS denotes offline Dynamics-Aware Discovery of Skills (Sharma et al., 2020a;b); and COMBO denotes Conservative Offline Model-Based Policy Optimization (Yu et al., 2021).

A. Appendix

Implementation

Networks

All MLPs use ReLU activations between hidden layers and output Gaussian distributions parameterized by a mean and a Softplus standard deviation, unless noted otherwise.

Abstract State Encoder: Observations are first embedded through a 2-layer MLP. The embeddings are concatenated with the previous action and fed into a 4-layer causal (unidirectional) GRU with hidden size h. $q_\phi(S_k)$ are read off at segment boundaries $t \in \{0, H, 2H, \ldots, KH-1\}$ through linear mean and Softplus sigma heads.

Skill Encoder: Observations in segment k are embedded with the same 2-layer MLP structure, concatenated with actions, and processed by a 4-layer bidirectional GRU with hidden size h, producing a $2h$-dimensional summary. The current abstract state S_k is embedded through a single linear layer and concatenated with the GRU output and then passed through linear mean and Softplus sigma heads.

Abstract Dynamics: A 2-layer MLP takes the concatenation of S_k and z_k followed by separate mean and sigma heads.

Segment Decoder: To give the decoder a smooth per-timestep context, S_k and S_{k+1} are linearly interpolated across the H timesteps of segment k. The interpolated state is concatenated with z_k, passed through a 2-layer MLP, followed by mean and sigma heads.

Observation Decoder: A 2-layer MLP followed by mean and sigma heads reconstructs the initial observation from the initial abstract state.

Skill Prior: A 2-layer MLP followed by mean and sigma heads predicts the prior distribution over skills conditioned on the current abstract state.

Segment Score Model: A 2-layer MLP takes the concatenation of S_k, z_k, and goal g followed by scalar mean and sigma heads. For AntMaze, the target score is the negative distance to the goal at the end of segment k.

Low-Level Policy: A 2-layer MLP takes the concatenation of the current observation and the skill embedding, followed by mean and Softplus sigma heads.

Training

Hyperparameters

Hyperparameter	Value
Model	
h	256
$\lvert\mathcal{S}\rvert$	32
$\lvert z\rvert$	256
H	10
K	4
Offline Training	
Optimizer	Adam
lr_E	5×10^{-5}
lr_M	5×10^{-5}
Batch size	100
Online Finetuning	
lr_E, lr_M	1×10^{-6}
Episodes	5,000
Gradient steps/episode	10
Offline batch ratio	0.8
Replay buffer capacity	5,000
CEM Planning	
Population size	100
Keep fraction	0.5
Iterations (collect)	100
Iterations (eval)	100
Skill sequence length	10

Table 4. Hyperparameters used for SALT on AntMaze-medium-diverse.