# FROM BIDS TO MODES: POLICY REFINEMENT WITH LATENT DECISION MODES FOR AUTO-BIDDING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Auto-bidding plays a central role in modern computational advertising, yet learning improved policies solely from offline trajectories remains challenging. Existing methods typically perform policy improvement directly in the original continuous bid space, which is only sparsely covered by training data. Moreover, bid-level optimization obscures distinctions among different decision tendencies, making it difficult to switch among them dynamically as market and budget conditions evolve. To address this, we introduce **PRISM** (**P**olicy **R**efinement with latent dec**IS**ion **M**odes), a framework for auto-bidding that optimizes policies over a compact mode-level decision space. To construct this space, PRISM uses return-tilted mode discovery to learn a partition that captures outcome-relevant distinctions between modes. Building on the discovered modes, it selects the best-evaluated mode from contextually supported candidates and realizes it through a continuous-bid policy trained with mode-referenced weighting. Experiments on AuctionNet show that PRISM consistently outperforms competitive auto-bidding baselines, while D4RL evaluations highlight its potential as a general policy-refinement framework for offline data containing diverse decision behaviors.

## 1 INTRODUCTION

Auto-bidding is a budget-constrained sequential decision problem at the core of modern advertising platforms (Cai et al., 2017; Wu et al., 2018; Su et al., 2024). Each bid affects not only the immediate probability and cost of winning an impression, but also, through its expenditure, the budget available for subsequent auctions (Cai et al., 2017; Su et al., 2024). As market conditions and budget availability evolve throughout a delivery period, effective bidding behavior must adapt accordingly (Su et al., 2024). However, online interaction in live advertising systems entails substantial risks and costs, motivating policy improvement from offline trajectories (Korenkevych et al., 2024; Gao et al., 2025). Offline policy improvement must therefore learn reliably from finite data while enabling bidding behavior to adapt to evolving environments.

Existing auto-bidding approaches address only part of this challenge. Generative methods model historical trajectory distributions but remain limited by the behaviors represented in fixed offline data (Guo et al., 2024; Gao et al., 2025; Mou et al., 2026), whereas value-based methods directly optimize long-term returns but are vulnerable to extrapolation errors under distribution shift (Fujimoto et al., 2019; Kumar et al., 2020). More broadly, finite data sparsely cover the continuous bid space, limiting reliable improvement beyond data support (Zhou et al., 2021). Meanwhile, when complementary bidding tendencies are represented only through their continuous action realizations, a unified policy may blend them into compromise bids rather than preserve them as alternatives for context-dependent switching.

Such variation in decision tendencies is evident in AuctionNet (Su et al., 2024). As shown in Figure 1, time-step-level decisions span a continuous landscape of relative bid intensities and market changes, summarized by six descriptive regimes. Individual campaigns traverse multiple regimes, while similar regimes recur across campaigns in different orders; 63% of campaigns change their dominant regime between the first and final thirds of their lifetimes. Crucially, these regimes represent recurring decision tendencies that manifest through heterogeneous bidding behaviors across varying auction contexts, with no single tendency dominating in long-term effectiveness.



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

Existing behavioral abstractions, whether options or skills, define their latent units by how behaviors are formed or how they differ from one another (Bacon et al., 2017; Eysenbach et al., 2018; Ajay et al., 2020; Mao et al., 2024; Wang et al., 2025). However, our AuctionNet analysis challenges this premise: behavioral form does not carry consistent decision-level semantics in auto-bidding. Identical bidding tendencies are realized through numerically distinct bids under different auction conditions, while similar bids may lead to fundamentally different long-term consequences. Evaluating and selecting among bidding decisions therefore requires units grounded in these long-term consequences rather than in behavioral form.

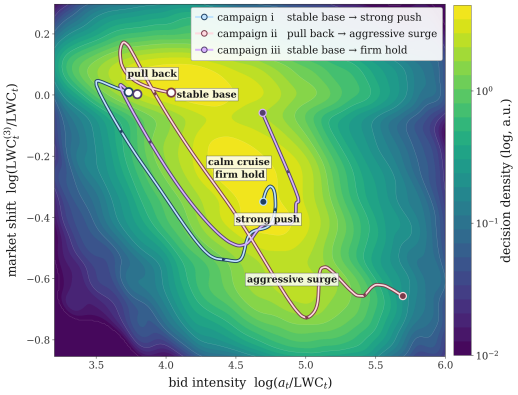


Figure 1: **Bidding regimes and trajectory dynamics in AuctionNet.** The background shows the log-density of time-step-level decisions. The horizontal axis is the bid intensity relative to the long-term winning-cost baseline $\mathrm{LWC}_t$; the vertical axis is the short-term market shift, where $\mathrm{LWC}_t^{(3)}$ is the recent three-step average winning cost. Unsupervised clustering identifies six descriptive regimes arranged along a diagonal from conservative to aggressive bidding, and three held-out trajectories illustrate transitions among them, with $t = 0$ and $t = T$ marking the first and last decisions of each trajectory. Trajectories transition among bidding regimes, which recur across trajectories.

We introduce *latent decision modes*, per-decision abstractions whose boundaries are shaped by return-tilted outcome information rather than behavioral criteria. To learn and exploit them, we propose **PRISM** (**P**olicy **R**efinement with latent dec**IS**ion **M**odes), a two-stage framework that structures offline auto-bidding policy improvement around these modes. In the first stage, a return-tilted information decomposition distinguishes the outcome-relevant variation captured between modes from that remaining within modes. Guided by this decomposition, a mode encoder and a contextual evaluator are trained alternately to discover the mode partition and estimate the outcome evidence of each mode under its surrounding context. In the second stage, a contextual support model identifies modes represented in the historical data, whose support-weighted evidence forms a reference for training a mode-conditioned continuous-bid policy through mode-referenced weighting. At deployment, PRISM retains the contextually supported modes, selects the best-evaluated candidate, and produces the corresponding bid.

Our contributions are summarized as follows. (1) We introduce **latent decision modes**, a per-decision abstraction that organizes recurring decision tendencies into a compact mode space, and derive a return-tilted information decomposition that yields a principled mode-discovery objective. (2) We propose **PRISM**, a framework that turns this decomposition into a trainable objective for mode discovery and couples it with mode-gated policy optimization. (3) PRISM consistently outperforms strong baselines on AuctionNet, with ablations tracing its gains to mode discovery and contextual switching, while D4RL results highlight its **broader applicability beyond auto-bidding**.

## 2 Preliminaries

### 2.1 Offline Reinforcement Learning

We consider an episodic Markov decision process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, in which a policy $\pi(a \mid s)$ interacts with the environment over a finite horizon $T$ to maximize the expected discounted return $J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{T-1} \gamma^t r_t\right]$. Instead of further interaction, offline policy improvement relies on a static dataset $\mathcal{D}$ collected by an unknown behavior policy $\pi_{\mathrm{beh}}$. The goal of offline policy improvement is to obtain a policy with higher return than $\pi_{\mathrm{beh}}$ using $\mathcal{D}$ alone.

A central difficulty is that the outcome of an action is observable only for the actions that actually appear in $\mathcal{D}$; for actions outside its support, value estimates must be extrapolated and are therefore unreliable (Fujimoto et al., 2019; Kumar et al., 2020). A standard formulation consequently

regularizes policy improvement toward the behavior policy:

$$\max_{\pi} \mathbb{E}_{s\sim\mathcal{D},\, a\sim\pi(\cdot|s)}[Q(s,a)] - \lambda\, \mathbb{E}_{s\sim\mathcal{D}}\left[D_{\mathrm{KL}}\big(\pi(\cdot \mid s)\,\|\,\pi_{\mathrm{beh}}(\cdot \mid s)\big)\right], \tag{1}$$

where $\lambda > 0$ controls the regularization strength. Its nonparametric solution takes the form

$$\pi^\star(a \mid s) \propto \pi_{\mathrm{beh}}(a \mid s) \exp\left(\frac{Q(s,a)}{\lambda}\right), \tag{2}$$

which can be viewed as an exponential tilting of the behavior policy toward actions with stronger long-term outcomes (Todorov, 2006; Kappen et al., 2012; Levine, 2018). Offline trajectories provide an observed measure of long-term outcome through the realized return-to-go $G_t = \sum_{\ell=t}^{T-1} \gamma^{\ell-t} r_\ell$, which records the rewards accumulated following each decision. We define the outcome evidence as $y_t = \exp(G_t/\lambda)$, where $\lambda > 0$ controls the strength of the return weighting.

### 2.2 Auto-Bidding as Sequential Decision-Making

In online advertising, an advertiser competes for a sequence of $N$ impression opportunities arriving within a delivery period. For each opportunity $i$, the advertiser submits a bid; the highest bidder wins the impression, obtains its value $v_i$, and pays the associated cost $c_i$. Let $o_i \in \{0, 1\}$ denote the winning indicator. The advertiser seeks to maximize the total value under a budget $B$ and additional performance constraints:

$$\max_{\{o_i\}} \sum_i o_i v_i \quad \text{s.t.} \quad \sum_i o_i c_i \le B, \qquad \frac{\sum_i c_{ij} o_i}{\sum_i p_{ij} o_i} \le C_j,\ \forall j, \tag{3}$$

where $c_{ij}$ and $p_{ij}$ denote the cost and the performance indicator (e.g., conversions) of impression $i$ with respect to constraint $j$, and $C_j$ is its threshold.

Under standard assumptions, the KKT conditions yield a value-proportional bidding rule

$$b_i^\star = \alpha_0 v_i + \sum_{j=1}^{J} \alpha_j p_{ij} C_j, \tag{4}$$

where $\alpha_0$ is the optimal bidding parameter and $\alpha_j$ are the Lagrange multipliers of the constraints (Zhang et al., 2014; He et al., 2021). In production, non-stationary competitors and fluctuating traffic render a fixed set of Lagrange multipliers suboptimal over the delivery period. Auto-bidding is therefore formulated as a sequential decision task adjusting them over time based on changing market conditions and historical feedback. At step $t$, the state $s_t$ describes the bidding environment, such as remaining budget, remaining time, and historical bidding outcomes; the action $a_t = (a_t^{\alpha_0}, \ldots, a_t^{\alpha_J})$ updates the bidding parameters, which determine the bid for each impression in that step through equation 4; and the reward $r_t = \sum_{i=1}^{M_t} o_i v_i$ is the total value of the $M_t$ impressions won during step $t$. The bidding procedure is then represented as a trajectory $\tau = (s_0, a_0, r_0, \ldots, s_{T-1}, a_{T-1}, r_{T-1})$, so that the dataset $\mathcal{D}$ in Section 2.1 takes exactly this form.

## 3 Methodology

### 3.1 Latent Decision Modes

PRISM organizes offline policy improvement around latent decision modes. Rather than learning directly in the continuous action space, it first discovers a compact set of modes that preserves the outcome-relevant distinctions among historical decisions, and then performs policy improvement over this mode space, where each mode is evaluated under its surrounding context and the policy selects among contextually supported modes before realizing them as continuous actions.

Formally, each decision is represented by a discrete latent variable $m_t \in \{1, \ldots, K\}$. A latent decision mode is a reusable decision pattern expressed by an individual state–action pair; modes may change within a trajectory and recur across different trajectories and contexts. A mode encoder $q_\theta(m_t \mid s_t, a_t)$ identifies the mode expressed by the current decision, and a context-conditioned evaluator $\mu_\phi(c_t, m)$ is trained to fit the conditional outcome evidence $\mathbb{E}_{\mathrm{beh}}[y_t \mid C_t = c_t, M_t = m]$,

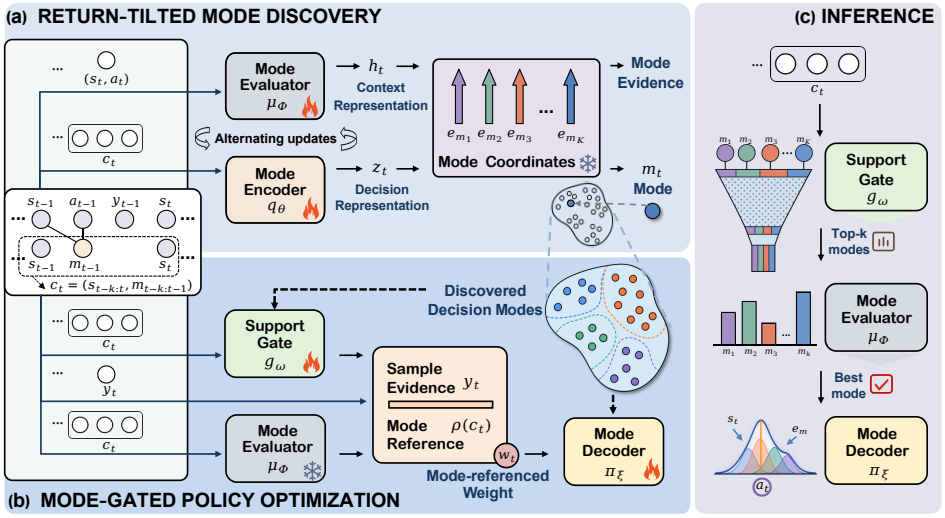


Figure 2: **Overview of PRISM.** (a) Return-tilted mode discovery alternately trains a mode encoder $q_\theta$ and a contextual evaluator $\mu_\phi$ over shared mode coordinates, yielding a set of discovered decision modes. (b) Mode-gated policy optimization combines a support gate $g_\omega$ and the evaluator $\mu_\phi$ into a mode reference $\rho(c_t)$, against which the sample evidence $y_t$ is compared to give the weight $y_t/\rho(c_t)$ for training the mode-conditioned decoder. (c) At inference, the top-$k$ supported modes are retained, the best-evaluated mode is selected, and the decoder outputs the continuous bid.

where $y_t = \exp(G_t/\lambda)$ is defined in Section 2.1. The context $c_t = (s_{t-k:t}, m_{t-k:t-1})$ contains the current and recent states together with the preceding modes. Thus, $(s_t, a_t)$ determines which mode is expressed, while $c_t$ determines how that mode is evaluated.

As illustrated in Figure 2, PRISM proceeds in two stages. *Return-Tilted Mode Discovery* (Section 3.2) alternately trains the encoder and the evaluator, so that the resulting partition preserves the decision information exposed by return tilting while each mode acquires a context-dependent evaluation. *Mode-Gated Policy Optimization* (Section 3.3) then converts these modes into an executable policy. For training, *Mode-Referenced Policy Improvement* (MRPI) weights each decision by $y_t/\rho(c_t)$, where $\rho(c_t)$ aggregates evaluator evidence weighted by a contextual support gate $g_\omega(m \mid c)$, so that a mode-conditioned decoder $\pi_\xi(a \mid s, e_m)$ learns the stronger realizations of each mode rather than their average. For inference, *Support-Prioritized Mode Gating* (SPMG) uses the same gate to retain the top-$k$ modes under $c_t$ and the evaluator to select the best among them, which the decoder then realizes as a continuous bid.

### 3.2 Return-Tilted Mode Discovery

Return-tilted mode discovery learns a latent partition that preserves distinctions exposed by return weighting. Return tilting first identifies outcome-relevant decision information and decomposes it into the information retained between modes and the residual variation within modes. This decomposition motivates minimizing the residual, leading to alternating updates between mode assignment and context-dependent evidence estimation. Both components are then parameterized over shared mode coordinates to maintain consistent latent identities throughout subsequent policy optimization.

**Return-tilted information decomposition.** For each step $t$, let $p_{\mathrm{beh},t}(c, a, y)$ be the joint distribution of the context, current action, and outcome evidence under the behavior policy. We define its time-step return-tilted counterpart as

$$p_{\mathrm{tilt},t}(c, a, y) = \frac{y\, p_{\mathrm{beh},t}(c, a, y)}{Z_t}, \qquad Z_t = \mathbb{E}_{\mathrm{beh}}[y_t]. \tag{5}$$

Compared with the behavior distribution, this transformation assigns more probability mass to decisions followed by larger returns. Conditioning equation 5 on $c_t$ gives

$$\frac{p_{\mathrm{tilt}}(a_t \mid c_t)}{p_{\mathrm{beh}}(a_t \mid c_t)} = \frac{\mathbb{E}_{\mathrm{beh}}[y_t \mid c_t, a_t]}{\mathbb{E}_{\mathrm{beh}}[y_t \mid c_t]}. \tag{6}$$

We omit the time subscript on the distributions when no ambiguity arises.

Fixing $c_t$, the current state $s_t$ is already observed, so the remaining decision variable is $a_t$. We attach the same stochastic mode encoder to the behavior and tilted distributions: $p_\bullet(a_t, m_t \mid c_t) = p_\bullet(a_t \mid c_t) q_\theta(m_t \mid s_t, a_t)$, where $\bullet \in \{\mathrm{beh}, \mathrm{tilt}\}$. Since the encoder is shared, it cancels in their joint density ratio. Applying the KL chain rule gives

$$\begin{aligned}\underbrace{D_{\mathrm{KL}}\left(p_{\mathrm{tilt}}(a_t \mid c_t) \,\|\, p_{\mathrm{beh}}(a_t \mid c_t)\right)}_{U(c_t)} = \underbrace{D_{\mathrm{KL}}\left(p_{\mathrm{tilt}}(m_t \mid c_t) \,\|\, p_{\mathrm{beh}}(m_t \mid c_t)\right)}_{S_q(c_t)} \\ + \underbrace{\mathbb{E}_{m \sim p_{\mathrm{tilt}}(\cdot \mid c_t)}\left[D_{\mathrm{KL}}\left(p_{\mathrm{tilt}}(a_t \mid c_t, m) \,\|\, p_{\mathrm{beh}}(a_t \mid c_t, m)\right)\right]}_{R_q(c_t)}.\end{aligned} \tag{7}$$

$U(c_t)$ is the action-level information exposed by return tilting, $S_q(c_t)$ is the part retained by the mode representation, and $R_q(c_t)$ is the information that remains among actions assigned to the same mode. Since $U(c_t)$ does not depend on $q_\theta$, maximizing $S_q(c_t)$ is equivalent to minimizing $R_q(c_t)$. Mode discovery therefore seeks a partition that captures return-related distinctions between modes rather than leaving them within a mode. The detailed derivation is in Appendix B.

**From residual minimization to mode learning.** Let $\bar{y}(c, a) = \mathbb{E}_{\mathrm{beh}}[y_t \mid c_t = c, a_t = a]$ and $\mu_q(c, m) = \mathbb{E}_{\mathrm{beh}}[y_t \mid c_t = c, m_t = m]$ under the joint distribution induced by $q_\theta$. Aggregating the conditional residual under the behavior distribution yields

$$R_{\mathrm{seq}}(q) = \mathbb{E}_{\mathrm{beh}}\left[\sum_t \sum_{m=1}^{K} q(m \mid s_t, a_t) d_{\mathrm{I}}\left(\bar{y}(c_t, a_t) \,\|\, \mu_q(c_t, m)\right)\right], \tag{8}$$

where $d_{\mathrm{I}}(u\|v) = u \log(u/v) - u + v$. Thus, minimizing the residual groups actions whose return-tilted evidence admits the same context-conditioned summary.

The practical objective follows directly from equation 8. We replace $\mu_q$ by the learned evaluator $\mu_\phi$ and use the observed $y_t$ as a sample estimate of $\bar{y}(c_t, a_t)$. The encoder minimizes

$$\mathcal{L}_{\mathrm{enc}}(\theta; \phi) = -\mathbb{E}_{t \sim \mathcal{D}}\left[\sum_{m=1}^{K} q_\theta(m \mid s_t, a_t)\left(y_t \log \mathrm{sg}[\mu_\phi(c_t, m)] - \mathrm{sg}[\mu_\phi(c_t, m)]\right)\right], \tag{9}$$

while the evaluator minimizes

$$\mathcal{L}_{\mathrm{eval}}(\phi; \theta) = -\mathbb{E}_{t \sim \mathcal{D}}\left[\sum_{m=1}^{K} \mathrm{sg}[q_\theta(m \mid s_t, a_t)]\left(y_t \log \mu_\phi(c_t, m) - \mu_\phi(c_t, m)\right)\right]. \tag{10}$$

For a fixed encoder, the population optimum is $\mu^\star(c, m) = \mathbb{E}_{\mathrm{beh}}[y_t \mid c_t = c, m_t = m]$. The two objectives are coordinate updates of the same form: the evaluator estimates the outcome evidence summarized by each mode, and the encoder assigns decisions to modes based on the frozen evaluator. Alternating these updates implements the residual minimization implied by equation 7.

**Shared mode coordinates.** The encoder and evaluator must refer to the same mode identities. We therefore represent the $K$ modes using a fixed orthonormal frame $E = [e_1, \ldots, e_K] \in \mathbb{R}^{d \times K}$ with $E^\top E = I_K$. The frame is initialized once by a thin QR decomposition of a Gaussian matrix and remains fixed throughout training.

The shared frame also gives a compact factorization of context-dependent mode evaluation. Instead of learning an unrelated scalar predictor for every mode, the context network produces one representation $h_\phi(c)$, and its projection onto $e_m$ gives the log-evidence of mode $m$. Specifically, with $\bar{z}_\theta(s, a) = z_\theta(s, a)/(\|z_\theta(s, a)\|_2 + \epsilon_z)$, we use

$$q_\theta(m \mid s, a) = \mathrm{softmax}_m\left(\frac{\bar{z}_\theta(s, a)^\top e_m}{\sigma}\right), \qquad \mu_\phi(c, m) = \exp\left(h_\phi(c)^\top e_m\right), \tag{11}$$

where $\sigma > 0$ is fixed. Normalizing $z_\theta$ makes mode assignment depend on its direction relative to the fixed frame. We leave $h_\phi(c)$ unnormalized because its magnitude is required to model the scale of the positive outcome evidence. Sharing $E$ aligns three operations: the encoder identifies a mode, the evaluator assigns its context-dependent outcome evidence, and the decoder realizes it as a continuous action.

### 3.3 Mode-Gated Policy Optimization

Mode discovery identifies a set of decision patterns and evaluates them under each context, but does not by itself produce an executable policy. Two additional operations are required: the policy must learn to generate continuous actions for each mode, and it must restrict mode selection to candidates that are well represented in the historical data under the current context. MGPO addresses these two operations through *Mode-Referenced Policy Improvement* (MRPI) and *Support-Prioritized Mode Gating* (SPMG), respectively.

After the first stage, we freeze $q_{\theta^\star}$ and $\mu_{\phi^\star}$. The trained encoder assigns a mode label $\hat{m}_t = \arg\max_m q_{\theta^\star}(m \mid s_t, a_t)$ to every sample in the dataset. These discovered labels determine the mode histories used by MGPO and provide supervision for the support gate.

#### 3.3.1 Mode-Referenced Policy Improvement

The support gate predicts which discovered modes are covered by the behavior data under context $c$. We parameterize it in the shared frame as $g_\omega(m \mid c) = \mathrm{softmax}_m(l_\omega(c)^\top e_m)$ and fit it to the discovered labels:

$$\mathcal{L}_{\mathrm{sup}}(\omega) = -\mathbb{E}_{t\sim\mathcal{D}}\left[\sum_{m=1}^{K} q_{\theta^\star}(m \mid s_t, a_t)\, \log g_\omega(m \mid c_t)\right], \qquad c_t = (s_{t-k:t},\, \hat{m}_{t-k:t-1}). \tag{12}$$

At the population optimum, $g_\omega(m \mid c)$ recovers the behavior distribution over modes under context.

MRPI uses the support gate to aggregate the frozen mode evaluations into a contextual reference:

$$\rho(c) = \sum_{m=1}^{K} g_\omega(m \mid c)\mu_{\phi^\star}(c, m). \tag{13}$$

When both models reach their population targets, the law of total expectation gives $\rho(c) = \mathbb{E}_{\mathrm{beh}}[y_t \mid c_t = c]$. The decoder weight is then

$$w_t = \mathrm{sg}\left[\mathrm{clip}\left(\frac{y_t}{\rho(c_t) + \epsilon}, w_{\min}, w_{\max}\right)\right]. \tag{14}$$

The decoder is a Gaussian policy $\pi_\xi(a \mid s, e_m) = \mathcal{N}(a; f_\xi(s, e_m), \mathrm{diag}(\varsigma_\xi^2(s, e_m)))$. More expressive conditional action models can also be used. MRPI minimizes

$$\mathcal{L}_{\mathrm{dec}}(\xi) = -\mathbb{E}_{t\sim\mathcal{D}}\left[w_t \log \pi_\xi(a_t \mid s_t, e_{\hat{m}_t})\right]. \tag{15}$$

The mode label specifies which decision pattern is realized, while the mode-referenced weight favors realizations with stronger outcome evidence than the behavior supported in the same context.

#### 3.3.2 Support-Prioritized Mode Gating

The support gate and evaluator play different roles at deployment. $g_\omega(m \mid c_t)$ defines which modes are eligible under the current context, while $\mu_{\phi^\star}(c_t, m)$ ranks their outcome evidence. SPMG first forms the support-prioritized candidate set and then selects its highest-evaluated member:

$$\mathcal{M}_{k_{\mathrm{sup}}}(c_t) = \underset{m}{\text{Top-k}}\, g_\omega(m \mid c_t), \qquad m_t^\star = \arg\max_{m\in\mathcal{M}_{k_{\mathrm{sup}}}(c_t)} \mu_{\phi^\star}(c_t, m). \tag{16}$$

This ordering helps reduce the risk of selecting weakly supported modes solely based on high evaluator outputs. The selected coordinate $e_{m_t^\star}$ is passed to the Gaussian decoder, which produces $\pi_\xi(\cdot \mid s_t, e_{m_t^\star})$. The selected mode is appended to the history used to construct $c_{t+1}$. The complete training and deployment procedures are provided in Appendix D.

# 4 EXPERIMENTS

## 4.1 EXPERIMENTAL SETUP

**Benchmarks.** We evaluate on two benchmarks. AuctionNet (Su et al., 2024) is a budget-constrained advertising-bidding benchmark in which a policy must allocate a limited budget across a sequence of auction opportunities; we report returns under budget levels from $50\%$ to $150\%$ of the reference budget, together with a sparse variant (AuctionNet-sparse) with substantially reduced conversion feedback. As a generalization test, we also evaluate on the D4RL Gym (Fu et al., 2020) benchmark, covering the medium (`-m`), medium-replay (`-mr`), and medium-expert (`-me`) datasets of HalfCheetah, Hopper, and Walker2d. We report scores on both benchmarks, with higher being better. Detailed evaluation criteria and experimental settings are provided in Appendix E.

**Baselines.** On AuctionNet, we compare PRISM against ten representative auto-bidding baselines: value-based offline RL (BCQ (Fujimoto et al., 2019), CQL (Kumar et al., 2020), IQL (Kostrikov et al., 2021)), bidding-specific constrained optimization (USCB (He et al., 2021)), return-conditioned sequence models and their constraint-/score-conditioned variants (DT (Chen et al., 2021), CDT (Liu et al., 2023), DT-score (Li et al., 2025)), and generative auto-bidding (DiffBid (Guo et al., 2024), GAS (Li et al., 2025), GAVE (Gao et al., 2025)). On D4RL, we compare against nine representative offline RL baselines: behavior-regularized policy optimization (BC (Bain & Sammut, 1995), AWAC (Nair et al., 2020), TD3+BC (Fujimoto & Gu, 2021)), Q-learning with conservative, implicit, in-context compositional, flow-based, and action-chunked variants (CQL, IQL, ICQL (Xu et al., 2026), FQL (Park et al., 2025), QC (Li et al., 2026a)), and sequence modeling (DT).

## 4.2 OVERALL PERFORMANCE

**Performance on the AuctionNet Benchmark.** As shown in Table 1, PRISM achieves the highest score in all ten settings. On AuctionNet, it improves over the strongest baseline, GAS, by $7.6\%$ on average, with the largest gain of $15.0\%$ under the tightest budget constraint. This suggests that mode-level refinement is particularly valuable when scarce budgets amplify the long-term consequences of bidding decisions. PRISM also maintains an average improvement of $5.5\%$ on AuctionNet-sparse, indicating that its advantage persists under limited feedback. Overall, organizing offline decisions into return-relevant modes enables more effective long-horizon budget allocation across varying budget and feedback conditions.

**Performance on the D4RL Benchmark.** Table 2 shows that PRISM remains competitive beyond auto-bidding, achieving the best score on four of the nine D4RL tasks and the second-best on two others. Its gains are particularly evident on Hopper and Walker2d, where it ranks first on both the medium and medium-replay datasets. PRISM also performs close to the strongest methods on the medium-expert tasks, ranking second on halfcheetah-me and hopper-me. The results are less pronounced on HalfCheetah and walker2d-me, suggesting that a compact discrete mode space is not uniformly advantageous when effective control depends more strongly on fine-grained continuous action variation. Overall, the D4RL results demonstrate that PRISM extends effectively beyond auto-bidding, achieving particularly strong performance in tasks where heterogeneous offline behaviors can be organized into reusable decision modes.

## 4.3 EMPIRICAL ANALYSIS

**Component ablations.** Table 3 validates the three complementary components of PRISM. Removing return-tilted mode discovery, which applies return weighting directly without introducing modes, causes the largest degradation, reducing the Dense and Sparse Scores from $384$ and $39.1$ to $304$ and $28.3$, respectively. This demonstrates the importance of constructing a return-relevant mode space over which to optimize. Without SPMG, the mode with the highest evaluator output is selected directly, reducing the

Table 3: Ablation study of PRISM.

| Method | Dense | | Sparse | |
|---|---|---|---|---|
| | Score ↑ | Conv. ↑ | Score ↑ | Conv. ↑ |
| PRISM | **384** | **419** | **39.1** | **40.4** |
| w/o Mode Discovery | 304 | 334 | 28.3 | 32.5 |
| w/o SPMG | 347 | 385 | 30.1 | 34.2 |
| w/o MRPI | 362 | 388 | 34.7 | 36.2 |

Table 1: Scores on AuctionNet and AuctionNet-sparse under different budget levels. Higher is better. We report scores (mean ±standard deviation) over 5 seeds. Bold denotes the best result, underlining the best baseline.

| Dataset | Budget | USCB | BCQ | CQL | IQL | DiffBid | DT | CDT | DT-score | GAS | GAVE | PRISM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AuctionNet | 50% | 86 | 190 | 113 | 164 | 54 | 191 | 174 | 178 | 193 | 176 | **222** ± 3.46 |
| | 75% | 135 | 259 | 139 | 232 | 100 | 265 | 242 | 268 | 287 | 245 | **309** ± 2.87 |
| | 100% | 157 | 321 | 171 | 281 | 152 | 329 | 326 | 334 | 359 | 303 | **384** ± 4.31 |
| | 125% | 220 | 379 | 201 | 355 | 193 | 396 | 378 | 395 | 409 | 357 | **434** ± 2.76 |
| | 150% | 281 | 429 | 238 | 401 | 234 | 450 | 433 | 441 | 461 | 404 | **471** ± 4.12 |
| AuctionNet-sparse | 50% | 11.5 | 17.7 | 12.8 | 16.5 | 9.9 | 14.8 | 11.2 | 17.7 | 18.4 | 17.9 | **19.5** ± 0.32 |
| | 75% | 14.9 | 24.6 | 16.7 | 22.1 | 15.4 | 22.9 | 18.0 | 25.9 | 27.5 | 25.5 | **29.3** ± 0.45 |
| | 100% | 17.5 | 31.1 | 22.2 | 30.0 | 19.5 | 29.6 | 31.2 | 33.2 | 36.1 | 31.6 | **39.1** ± 0.82 |
| | 125% | 26.7 | 34.2 | 28.6 | 37.1 | 25.3 | 34.3 | 31.7 | 39.6 | 40.0 | 34.4 | **41.5** ± 0.67 |
| | 150% | 31.3 | 37.9 | 35.8 | 43.1 | 30.8 | 44.5 | 39.1 | 44.7 | 46.5 | 40.7 | **47.9** ± 0.41 |

Table 2: Normalized scores on the D4RL Gym benchmark. Higher is better. We report normalized scores over 5 seeds. Bold denotes the best result and underlining the second-best result in each row.

| Dataset | BC | AWAC | TD3+BC | DT | CQL | IQL | FQL | QC | ICQL | PRISM |
|---|---|---|---|---|---|---|---|---|---|---|
| halfcheetah-m | 42.6 | 43.5 | 43.1 | 42.8 | 44.4 | 42.5 | 43.9 | **48.2** | 45.9 | 44.2 ± 0.2 |
| hopper-m | 52.9 | 57.0 | 53.5 | 57.1 | 58.0 | 63.3 | 55.6 | 64.6 | 62.6 | **71.3** ± 2.1 |
| walker2d-m | 75.3 | 72.4 | 77.0 | 70.2 | 79.2 | 71.5 | 72.4 | 34.1 | 80.3 | **81.5** ± 1.1 |
| halfcheetah-me | 55.2 | 42.8 | 62.8 | 70.8 | 62.4 | 83.4 | 92.9 | **94.2** | 89.1 | 93.4 ± 0.2 |
| hopper-me | 52.5 | 55.8 | 78.2 | 57.5 | 105.4 | 98.5 | 60.1 | 44.0 | **111.0** | 110.6 ± 0.7 |
| walker2d-me | 107.5 | 74.5 | 109.2 | 70.7 | 98.7 | 109.8 | 101.0 | 102.8 | **113.3** | 108.5 ± 0.1 |
| halfcheetah-mr | 36.6 | 40.5 | 41.8 | 39.5 | **45.5** | 38.9 | 40.0 | 40.5 | 44.7 | 42.4 ± 0.4 |
| hopper-mr | 18.1 | 37.2 | 59.4 | 65.8 | 95.0 | 82.4 | 55.0 | 18.6 | 96.4 | **97.2** ± 1.0 |
| walker2d-mr | 26.0 | 27.0 | 41.5 | 54.8 | 77.2 | 61.0 | 60.9 | 46.6 | 81.9 | **83.9** ± 1.1 |
| **Average** | 51.9 | 50.1 | 62.9 | 58.8 | 74.0 | 72.4 | 64.6 | 54.8 | 80.6 | **81.4** |

scores to 347 and 30.1; this confirms that mode evaluation must be constrained by contextual support. Without MRPI, the decoder is trained without mode-referenced weighting, yielding scores of 362 and 34.7 and showing that the selected mode benefits from weighted learning of its continuous-action realization. The consistent conversion trends reinforce this division of roles: mode discovery determines *what* to optimize, SPMG determines *which* mode to select, and MRPI determines *how* the selected mode is realized as an action.

**Mode specialization and contextual selection.** Figure 3 examines whether the gain of PRISM comes from one dominant mode or from contextual mode selection. We compare the complete method with policies that execute a single fixed mode throughout each episode. None of the eight fixed modes approaches PRISM: the best fixed mode achieves a score of 324, compared with 384 for the complete method (Figure 3(a)). Their reward and budget trajectories further reveal distinct bidding characteristics (Figures 3(b)–(c)). Some modes spend aggressively and exhaust the budget early, whereas others conserve budget but accumulate reward slowly. Hence, the learned modes are complementary behavioral patterns, not candidates ordered by intrinsic quality. By switching modes with context, PRISM regulates the spending pace and preserves budget for valuable later opportunities. This confirms that its advantage arises from the selection mechanism rather than from discovering a single universally superior mode.

**Comparison with latent abstractions.** We compare PRISM with LOM, which models state-conditioned behavior through Gaussian action modes (Wang et al., 2025), and LPT, which encodes each trajectory into a persistent latent plan (Kong et al., 2024). As shown in Figure 4, both recover the coarse value gradient but provide virtually no additional resolution beyond the context-only baseline, changing the Pearson correlation with within-trajectory conversion percentiles by only $+0.015$ and $-0.001$, respectively. LOM reflects action-distribution multimodality rather than the long-term roles of individual decisions, while LPT smooths over distinct roles within a trajectory. In contrast, PRISM captures finer local variations in these observed outcomes, increasing the correlation by +0.124. This advantage is further reflected in AuctionNet scores of 286, 301, and 384 for LOM, LPT, and PRISM, respectively. More details are provided in Appendix C.2.

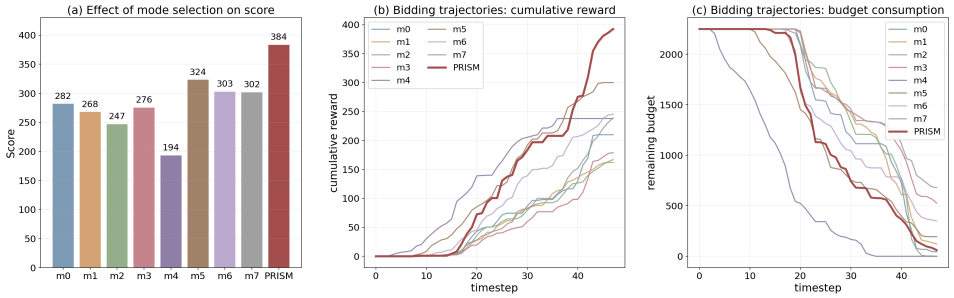


Figure 3: **Effect of contextual mode selection.** (a) Score and total reward (conversions) of PRISM and policies that execute a single fixed mode throughout an episode. (b) Cumulative-reward trajectories. (c) Remaining-budget trajectories. Individual modes exhibit distinct spending behaviors, whereas PRISM improves performance by switching among complementary modes according to context.

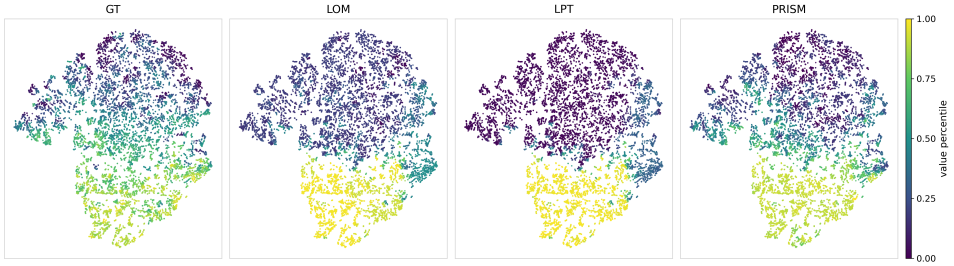


Figure 4: **Cross-fitted reconstruction of the decision-value field on AuctionNet.** Shared PCA–t-SNE coordinates are computed from standardized states, and GT is the within-trajectory percentile rank of each decision by observed conversions. K-means defines three common context bins. After splitting the data in half, each decision is assigned the mean GT percentile of samples in the opposite split with the same context bin and mode label. PRISM most closely reconstructs the GT field, preserving finer local value variations than LOM and LPT.

**Effect of key hyperparameters.** Figure 5 analyzes the support-set size $k_{\text{sup}}$, history-window size $H$, and number of modes $K$ on the AuctionNet dataset. Performance peaks at $k_{\text{sup}} = 4$: a smaller support set limits contextual mode selection, whereas a larger one admits less-supported candidates. Incorporating recent states and mode history consistently outperforms the history-free setting ($H = 0$), with the best results at $H = 4$; the slight decline at $H = 5$ indicates diminishing returns from longer histories. Increasing $K$ improves performance up to $K = 8$. Fewer modes underrepresent behavioral diversity, whereas additional modes fragment the data and weaken support and outcome-evidence estimation.

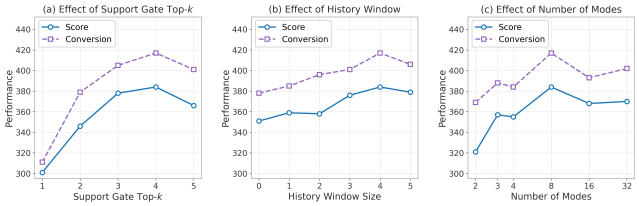


Figure 5: **Analysis of key hyperparameters in PRISM.** (a) Support-set size $k_{\text{sup}}$. (b) History-window size $H$. (c) Number of latent modes $K$.

## 5 Conclusion

We introduced PRISM, which reframes continuous-action policy improvement as optimization over reusable latent decision modes. By combining return-tilted mode discovery with mode-gated policy optimization, PRISM learns outcome-relevant modes, evaluates them in context, and decodes selected modes into continuous actions. Experiments support its effectiveness; future work will refine these mechanisms and extend the framework to broader sequential decision-making tasks.

### AI USE STATEMENT

In this work, we employed generative AI tools to refine the manuscript's language, correct grammatical errors, enhance overall readability, and make minor adjustments to LaTeX formatting. We also utilized these tools for limited programming assistance, specifically for debugging syntax errors and improving code clarity. We did not employ generative AI tools for creating datasets, developing the core methodology, deriving mathematical claims, running experiments, or analyzing experimental outcomes. All AI-generated text and code suggestions underwent thorough review, revision, and verification by the authors. The authors assume full responsibility for the final content of this work, including all claims, experimental results, and artifacts.

### REPRODUCIBILITY STATEMENT

The derivation of our method is provided in Appendix B; experimental details, including architectures, main hyperparameters, and dataset information, are provided in Appendix E. Our source code cannot be released at submission time because it is still undergoing a privacy and disclosure audit at our institution. If the reviewers find the code helpful for their evaluation, we will provide the code to them during the rebuttal period.

## REFERENCES

Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. Opal: Offline primitive discovery for accelerating offline reinforcement learning. *arXiv preprint arXiv:2010.13611*, 2020.

Marvin Alles, Philip Becker-Ehmck, Patrick van der Smagt, and Maximilian Karl. Constrained latent action policies for model-based offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37:70381–70405, 2024.

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine intelligence 15*, pp. 103–129, 1995.

Han Cai, Kan Ren, Weinan Zhang, Kleanthis Malialis, Jun Wang, Yong Yu, and Defeng Guo. Real-time bidding by reinforcement learning in display advertising. In *Proceedings of the tenth ACM international conference on web search and data mining*, pp. 661–670, 2017.

Leng Cai, Junxuan He, Yikai Li, Junjie Liang, Yuanping Lin, Ziming Quan, Yawen Zeng, and Jin Xu. Rtbagent: A llm-based agent system for real-time bidding. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 104–113, 2025.

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.

Xi Chen, Ali Ghadirzadeh, Tianhe Yu, Jianhao Wang, Alex Yuan Gao, Wenzhe Li, Liang Bin, Chelsea Finn, and Chongjie Zhang. Lapo: Latent-variable advantage-weighted policy optimization for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 36902–36913, 2022.

Alberto Silvio Chiappa, Briti Gangopadhyay, Zhao Wang, and Shingo Takamatsu. Auto-bidding in real-time auctions via oracle imitation learning. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 345–356, 2025.

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.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Jingtong Gao, Yewen Li, Shuai Mao, Peng Jiang, Nan Jiang, Yejing Wang, Qingpeng Cai, Fei Pan, Peng Jiang, Kun Gai, et al. Generative auto-bidding with value-guided explorations. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 244–254, 2025.

Jiayan Guo, Yusen Huo, Zhilin Zhang, Tianyu Wang, Chuan Yu, Jian Xu, Bo Zheng, and Yan Zhang. Generative auto-bidding via conditional diffusion modeling. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5038–5049, 2024.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

Yue He, Xiujun Chen, Di Wu, Junwei Pan, Qing Tan, Chuan Yu, Jian Xu, and Xiaoqiang Zhu. A unified solution to constrained bidding in online display advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2993–3001, 2021.

Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

Hilbert J Kappen, Vicenç Gómez, and Manfred Opper. Optimal control as a graphical model inference problem. *Machine learning*, 87(2):159–182, 2012.

Donghyeon Ki, JunHyeok Oh, Seong-Woong Shim, and Byung-Jun Lee. Prior-guided diffusion planning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 38:59790–59820, 2026.

Deqian Kong, Dehong Xu, Minglu Zhao, Bo Pang, Jianwen Xie, Andrew Lizarraga, Yuhao Huang, Sirui Xie, and Ying Nian Wu. Latent plan transformer for trajectory abstraction: Planning as latent space inference. *Advances in Neural Information Processing Systems*, 37:123379–123401, 2024.

Dmytro Korenkevych, Frank Cheng, Artsiom Balakir, Alex Nikulkov, Lingnan Gao, Zhihao Cen, Zuobing Xu, and Zheqing Zhu. Offline reinforcement learning for optimizing production bidding policies. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5251–5259, 2024.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Qiyang Li, Zhiyuan Paul Zhou, and Sergey Levine. Reinforcement learning with action chunking. *Advances in Neural Information Processing Systems*, 38:55518–55553, 2026a.

Wenhao Li. Efficient planning with latent diffusion. In *International Conference on Learning Representations*, volume 2024, pp. 40835–40867, 2024.

Yewen Li, Shuai Mao, Jingtong Gao, Nan Jiang, Yunjian Xu, Qingpeng Cai, Fei Pan, Peng Jiang, and Bo An. Gas: Generative auto-bidding with post-training search. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 315–324, 2025.

Yewen Li, Zhiyi Lyu, Peng Jiang, Qingpeng Cai, Fei Pan, Bo An, and Peng Jiang. Lbm: Hierarchical large auto-bidding model via reasoning and acting. In *Proceedings of the ACM Web Conference 2026*, pp. 75–86, 2026b.

Zuxin Liu, Zijian Guo, Yihang Yao, Zhepeng Cen, Wenhao Yu, Tingnan Zhang, and Ding Zhao. Constrained decision transformer for offline safe reinforcement learning. In *International conference on machine learning*, pp. 21611–21630. PMLR, 2023.

Yihuan Mao, Chengjie Wu, Xi Chen, Hao Hu, Ji Jiang, Tianze Zhou, Tangjie Lv, Changjie Fan, Zhipeng Hu, Yi Wu, et al. Stylized offline reinforcement learning: Extracting diverse high-quality behaviors from heterogeneous datasets. In *International Conference on Learning Representations*, volume 2024, pp. 51217–51237, 2024.

Zhiyu Mou, Yiqin Lv, Miao Xu, Qi Wang, Yixiu Mao, Jinghao Chen, Qichen Ye, Chao Li, Rongquan Bai, Chuan Yu, et al. Enhancing generative auto-bidding with offline reward evaluation and policy search. In *International Conference on Learning Representations*, volume 2026, pp. 17644–17675, 2026.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. *arXiv preprint arXiv:2502.02538*, 2025.

Yunshan Peng, Wenzheng Shu, Jiahao Sun, Yanxiang Zeng, Jinan Pang, Wentao Bai, Yunke Bai, Xialong Liu, and Peng Jiang. Expert-guided diffusion planner for auto-bidding. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 5963–5970, 2025.

Karl Pertsch, Youngwoon Lee, and Joseph Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on robot learning*, pp. 188–204. PMLR, 2021.

Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. In *Learning for dynamics and control*, pp. 1154–1168. PMLR, 2021.

Erick Rosete-Beas, Oier Mees, Gabriel Kalweit, Joschka Boedecker, and Wolfram Burgard. Latent plans for task-agnostic offline reinforcement learning. In *Conference on Robot Learning*, pp. 1838–1849. PMLR, 2023.

Kefan Su, Yusen Huo, Zhilin Zhang, Shuai Dou, Chuan Yu, Jian Xu, Zongqing Lu, and Bo Zheng. Auctionnet: A novel benchmark for decision-making in large-scale games. *Advances in Neural Information Processing Systems*, 37:94428–94452, 2024.

Emanuel Todorov. Linearly-solvable markov decision problems. *Advances in neural information processing systems*, 19, 2006.

Mianchu Wang, Yue Jin, and Giovanni Montana. Learning on one mode: Addressing multi-modality in offline reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 76423–76440, 2025.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

Di Wu, Xiujun Chen, Xun Yang, Hao Wang, Qing Tan, Xiaoxun Zhang, Jian Xu, and Kun Gai. Budget constrained bidding by model-free reinforcement learning in display advertising. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 1443–1451, 2018.

Qiushui Xu, Yuhao Huang, Yushu Jiang, Wenliang Zheng, Lei Song, Jinyu Wang, and Jiang Bian. in-context compositional q-learning for offline reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 38350–38379, 2026.

Weinan Zhang, Shuai Yuan, and Jun Wang. Optimal real-time bidding for display advertising. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1077–1086, 2014.

Wenxuan Zhou, Sujay Bajracharya, and David Held. Plas: Latent action space for offline reinforcement learning. In *Conference on Robot Learning*, pp. 1719–1735. PMLR, 2021.

# APPENDIX

## A RELATED WORK

### A.1 AUTO-BIDDING METHODS

Early auto-bidding methods formulated budget-constrained optimization through Lagrangian duality, yielding closed-form bidding functions (Zhang et al., 2014). Their limited adaptability to evolving auction dynamics motivated sequential formulations based on reinforcement learning (Cai et al., 2017; Wu et al., 2018), including USCB, which learns Lagrange multipliers for heterogeneous budget and cost constraints (He et al., 2021). As offline RL remains vulnerable to bootstrapping and extrapolation errors, recent work has shifted to generative modeling, recasting bidding as conditional trajectory generation with diffusion models (Guo et al., 2024; Peng et al., 2025). To overcome the imitation ceiling of purely conditional generation, subsequent work injects value signals through value-guided exploration (Gao et al., 2025), post-training search (Li et al., 2025), generative planning coupled with policy optimization (Mou et al., 2026), and imitation of hindsight-optimal trajectories (Chiappa et al., 2025). More recently, LLMs have been employed either as bidding agents with memory and reflection (Cai et al., 2025), or as high-level planners that condition downstream bid decisions (Li et al., 2026b).

Despite increasingly flexible policy models, existing approaches mainly improve decisions in the original action or trajectory space, without explicitly identifying recurring decision tendencies. Complementary behaviors in offline data are thus often blended into a single policy or confined to entire trajectories, rather than being reused as transferable units for context-dependent selection. PRISM instead organizes heterogeneous offline decisions into return-relevant, per-decision modes and performs policy refinement over this compact mode space.

### A.2 LATENT STRUCTURE IN OFFLINE RL

Offline RL mitigates distribution shift by restricting policy improvement to dataset-supported behavior, using, for example, behavior-aware or conservative value learning (Fujimoto et al., 2019; Kumar et al., 2020) and pessimistic dynamics models (Rafailov et al., 2021). Latent-variable methods enable supported improvement in a compact space rather than the raw action space (Zhou et al., 2021; Chen et al., 2022). Related work learns temporally extended latent skills or plans from unlabeled data for hierarchical control (Ajay et al., 2020; Pertsch et al., 2021; Rosete-Beas et al., 2023). Trajectory-level generative models use latents to preserve behavioral multimodality while enabling value-guided improvement (Chen et al., 2021; Janner et al., 2022; Wang et al., 2022; Hansen-Estruch et al., 2023); subsequent methods move generation and optimization into compact latent spaces to reduce cost and mode averaging (Li, 2024; Alles et al., 2024; Ki et al., 2026).

These methods primarily organize supported behavior by reconstruction, temporal, or generative criteria, but do not explicitly target decision-level distinctions in long-term return. In bidding, superficially similar bid sequences can yield very different outcomes through auction results, winning costs, and budget pacing. Credit assignment across auctions therefore requires separating such outcome-distinct modes. PRISM instead uses return-tilted outcome information to discover per-decision modes and refines the policy in the resulting compact, outcome-relevant space.

## B DETAILED DERIVATIONS

### B.1 RETURN-TILTED CONDITIONAL DISTRIBUTIONS AND KL DECOMPOSITION

Fix a time step $t$ and the construction of $c_t$, including its preceding mode assignments. We suppress the time subscript on distributions. Let $y_t = \exp(G_t/\lambda) > 0$ and assume $Z_t = \mathbb{E}_{\mathrm{beh}}[y_t] < \infty$. Define

$$\bar{y}(c,a) := \mathbb{E}_{\mathrm{beh}}[y_t \mid c_t = c, a_t = a], \qquad \rho_q(c) := \mathbb{E}_{\mathrm{beh}}[y_t \mid c_t = c] = \int p_{\mathrm{beh}}(a \mid c)\bar{y}(c,a)\, da. \tag{17}$$

The subscript $q$ indicates that the mode history in $c_t$ is generated by the encoder.

Marginalizing $y$ in equation 5, followed by $a$, gives

$$p_{\mathrm{tilt}}(c,a) = \frac{p_{\mathrm{beh}}(c,a)\bar{y}(c,a)}{Z_t}, \qquad p_{\mathrm{tilt}}(c) = \frac{p_{\mathrm{beh}}(c)\rho_q(c)}{Z_t}. \tag{18}$$

Hence, for almost every $c$ under the behavior context distribution,

$$p_{\mathrm{tilt}}(a \mid c) = p_{\mathrm{beh}}(a \mid c)\frac{\bar{y}(c,a)}{\rho_q(c)}, \tag{19}$$

which is equation 6.

Now attach the same encoder to both distributions: $p_\bullet(a, m \mid c) = p_\bullet(a \mid c)q_\theta(m \mid s, a)$, where $\bullet \in \{\mathrm{beh}, \mathrm{tilt}\}$ and $s = s_t$ is contained in $c$. On the tilted support,

$$\frac{p_{\mathrm{tilt}}(a, m \mid c)}{p_{\mathrm{beh}}(a, m \mid c)} = \frac{p_{\mathrm{tilt}}(a \mid c)q_\theta(m \mid s, a)}{p_{\mathrm{beh}}(a \mid c)q_\theta(m \mid s, a)} = \frac{p_{\mathrm{tilt}}(a \mid c)}{p_{\mathrm{beh}}(a \mid c)}. \tag{20}$$

Taking the expectation of its logarithm under $p_{\mathrm{tilt}}(a, m \mid c)$ and marginalizing $m$ yields

$$D_{\mathrm{KL}}(p_{\mathrm{tilt}}(a, m \mid c) \,\|\, p_{\mathrm{beh}}(a, m \mid c)) = D_{\mathrm{KL}}(p_{\mathrm{tilt}}(a \mid c) \,\|\, p_{\mathrm{beh}}(a \mid c)). \tag{21}$$

Alternatively, factoring the joint distributions by their mode marginals gives

$$\begin{aligned} &D_{\mathrm{KL}}(p_{\mathrm{tilt}}(a, m \mid c) \,\|\, p_{\mathrm{beh}}(a, m \mid c)) \\ &= \sum_m \int p_{\mathrm{tilt}}(a, m \mid c) \log \frac{p_{\mathrm{tilt}}(m \mid c)p_{\mathrm{tilt}}(a \mid c, m)}{p_{\mathrm{beh}}(m \mid c)p_{\mathrm{beh}}(a \mid c, m)}\, da \\ &= D_{\mathrm{KL}}(p_{\mathrm{tilt}}(m \mid c) \,\|\, p_{\mathrm{beh}}(m \mid c)) \\ &\quad + \mathbb{E}_{m \sim p_{\mathrm{tilt}}(\cdot \mid c)}[D_{\mathrm{KL}}(p_{\mathrm{tilt}}(a \mid c, m) \,\|\, p_{\mathrm{beh}}(a \mid c, m))]. \end{aligned} \tag{22}$$

Combining equation 21 and equation 22 gives $U(c) = S_q(c) + R_q(c)$, as stated in equation 7.

### B.2 Derivation of the mode-discovery objective

We express the residual in equation 7 as a behavior-distribution objective. The context construction is held fixed as in Appendix B.1, and all expectations below are assumed finite. Time subscripts on conditional quantities are omitted; each quantity is evaluated at the corresponding time step.

Under the joint distribution $p_{\mathrm{beh}}(c, a, y)q_\theta(m \mid s, a)$, the mean evidence within mode $m$ is

$$\mu_q(c, m) := \mathbb{E}_{\mathrm{beh}}[y_t \mid c_t = c, m_t = m] = \frac{\int p_{\mathrm{beh}}(a \mid c)q_\theta(m \mid s, a)\bar{y}(c, a)\, da}{\int p_{\mathrm{beh}}(a \mid c)q_\theta(m \mid s, a)\, da}. \tag{23}$$

Only modes with positive conditional probability are considered. By equation 19, the within-mode density ratio satisfies

$$\frac{p_{\mathrm{tilt}}(a \mid c, m)}{p_{\mathrm{beh}}(a \mid c, m)} = \frac{\bar{y}(c, a)}{\mu_q(c, m)}. \tag{24}$$

Moreover, the mode means preserve the context mean:

$$\mathbb{E}_{a \sim p_{\mathrm{beh}}(\cdot \mid c)}\left[\sum_m q_\theta(m \mid s, a)\mu_q(c, m)\right] = \rho_q(c) = \mathbb{E}_{a \sim p_{\mathrm{beh}}(\cdot \mid c)}[\bar{y}(c, a)]. \tag{25}$$

Thus, the terms $-u + v$ in $d_I(u\|v) = u\log(u/v) - u + v$ cancel under the encoder-weighted behavior expectation.

Using the shared encoder and equation 24, we obtain

$$\begin{aligned} \rho_q(c)R_q(c) &= \rho_q(c)\,\mathbb{E}_{(a,m) \sim p_{\mathrm{tilt}}(\cdot, \cdot \mid c)}\left[\log \frac{\bar{y}(c, a)}{\mu_q(c, m)}\right] \\ &= \mathbb{E}_{a \sim p_{\mathrm{beh}}(\cdot \mid c)}\left[\sum_m q_\theta(m \mid s, a)\bar{y}(c, a) \log \frac{\bar{y}(c, a)}{\mu_q(c, m)}\right] \\ &= \mathbb{E}_{a \sim p_{\mathrm{beh}}(\cdot \mid c)}\left[\sum_m q_\theta(m \mid s, a) d_I(\bar{y}(c, a) \| \mu_q(c, m))\right]. \end{aligned} \tag{26}$$

Finally, conditioning each time-step contribution in equation 8 on $c_t$ and applying equation 26 gives

$$R_{\text{seq}}(q) = \mathbb{E}_{\text{beh}}\left[\sum_t \sum_m q_\theta(m \mid s_t, a_t) d_I(\bar{y}(c_t, a_t) \| \mu_q(c_t, m))\right] = \sum_t \mathbb{E}_{\text{beh}}\left[\rho_q(c_t) R_q(c_t)\right]. \tag{27}$$

Hence, equation 8 aggregates the conditional KL residuals with context-dependent evidence weights $\rho_q(c_t)$. To obtain the training losses, replace $\mu_q$ by a positive evaluator $\mu_\phi$ and expand the per-decision discrepancy:

$$\sum_m q_\theta(m \mid s, a) d_I(\bar{y}(c, a) \| \mu_\phi(c, m)) = \bar{y}(c, a) \log \bar{y}(c, a) - \bar{y}(c, a) - \sum_m q_\theta(m \mid s, a)\left[\bar{y}(c, a) \log \mu_\phi(c, m) - \mu_\phi(c, m)\right]. \tag{28}$$

The first two terms are independent of the encoder and evaluator parameters under the fixed context construction. For the remaining terms, the definition of $\bar{y}(c, a)$ yields

$$\mathbb{E}_{\text{beh}}\left[y_t \log \mu_\phi(c, m) - \mu_\phi(c, m) \mid c_t = c, a_t = a\right] = \bar{y}(c, a) \log \mu_\phi(c, m) - \mu_\phi(c, m). \tag{29}$$

Since $q_\theta(m \mid s, a)$ is fixed given $(c, a)$, this identity also holds after weighting and summing over modes. Replacing $\bar{y}(c_t, a_t)$ with the observed $y_t$ therefore preserves the parameter-dependent population objective exactly.

# C Additional Results

## C.1 Training Stability Analysis

### C.1.1 Convergence of the Training Loss

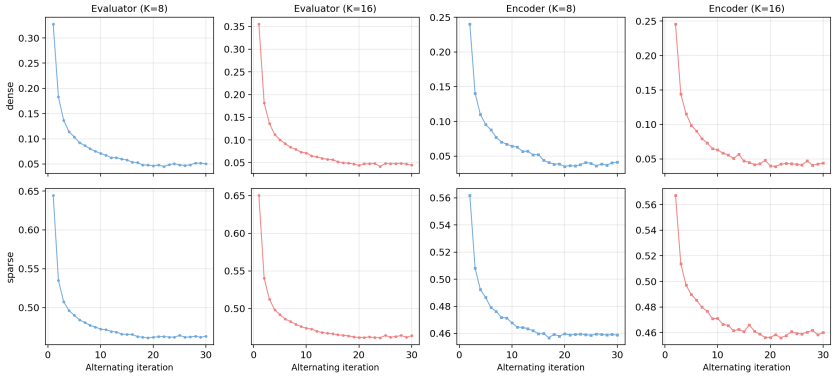


Figure 6: **Empirical convergence of mode discovery.** Rows correspond to dense and sparse AuctionNet data. Columns show the evaluator and encoder losses for $K \in \{8, 16\}$. Both losses decrease rapidly and reach stable plateaus, indicating empirically stable optimization behavior.

Figure 6 shows the optimization trajectories of mode discovery across different capacities. For both dense and sparse data, the evaluator and encoder losses decrease rapidly during the initial EM iterations and stabilize after approximately 15–20 iterations, with only minor fluctuations thereafter. The two objectives also reach their plateaus at similar rates, supporting the empirical convergence of the coupled mode assignment and evidence estimation procedure.

### C.1.2 Mode utilization and collapse

Figure 7 shows that increasing the nominal number of modes does not necessarily increase the effective abstraction capacity. For LOM, the number of collapsed components grows rapidly with

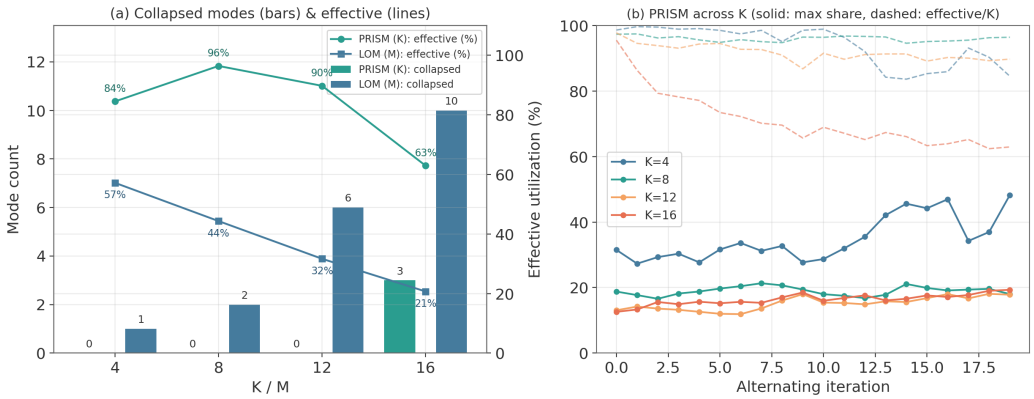


Figure 7: **Mode collapse and utilization across latent capacities.** (a) Bars show the number of collapsed modes (those assigned to fewer than $1\%$ of samples) for LOM and PRISM, while lines show normalized effective utilization $K_{\text{eff}}/K$, where $K_{\text{eff}} = \exp(H(p))$ and $H(p)$ is the entropy of the mode assignment proportions. LOM increasingly fails to convert additional components into effective capacity, whereas PRISM fully utilizes its modes up to $K = 12$. (b) PRISM assignment dynamics during alternating training. Solid curves denote the largest mode share and dashed curves denote normalized effective utilization. The stable trajectories show that the utilization gap in (a) is not caused by transient assignments.

$M$, reaching 10 out of 16, while effective utilization falls to $21\%$. Thus, most additional Gaussian components fail to represent distinct decision patterns. In contrast, PRISM activates all modes up to $K = 12$ and maintains $84\%$–$96\%$ effective utilization. This difference reflects their learning criteria: LOM identifies components from multimodality in the behavior-action distribution, where additional components can become redundant under uneven or overlapping action density; PRISM separates decisions when doing so reduces unexplained return-relevant variation within modes. At $K = 16$, PRISM also begins to exhibit redundant modes, suggesting that the specified capacity exceeds the distinctions supported by the data. Its strong utilization at smaller $K$ is therefore not imposed by uniform balancing, but emerges from the return-relevant structure captured by the modes.

## C.2 Analysis of Mode Structure and Dynamics

### C.2.1 Comparison of Mode Partitioning Schemes

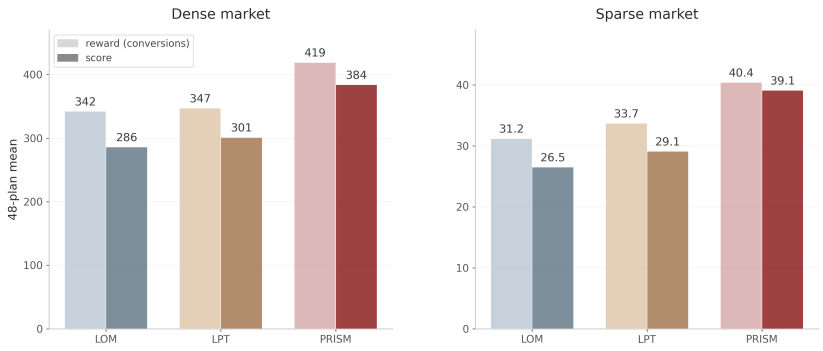


Figure 8: **Performance comparison of LOM, LPT, and PRISM on AuctionNet.** PRISM consistently achieves the best score and conversion performance across both settings.

Figure 8 compares the downstream performance of different latent abstractions. PRISM consistently outperforms LOM and LPT on both dense and sparse AuctionNet settings across score and

conversion metrics. These results show that PRISM's advantage extends beyond representation quality to policy performance.

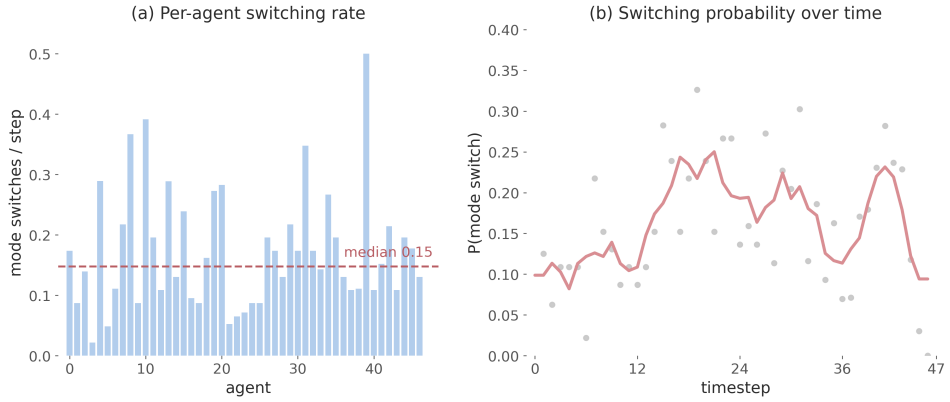


Figure 9: **Temporal mode-selection dynamics during AuctionNet evaluation rollouts on** 48 **dense-market trajectories.** (a) Per-agent switching rate, defined as the fraction of adjacent active steps with different selected modes; the dashed line marks the median of 0.15. (b) Fraction of active agents switching modes at each timestep. Gray points show the observed fractions and the red curve their five-step moving average. Switching is relatively infrequent initially, concentrates in the middle of the trajectory, and subsequently declines.

**Temporal dynamics of mode selection.** Figure 9 shows that mode selection is neither fixed throughout a trajectory nor changed indiscriminately at every step. Across the 48 evaluation agents, the median switching rate is 0.15, corresponding to approximately one switch every 6–7 active steps, while the substantial variation across agents reflects adaptation to their individual budget and market states. Temporally, switching is limited during the initial mode establishment, becomes more frequent in the middle of the trajectory as conditions evolve, and declines toward the end. These results reveal a structured pattern of persistent mode execution interleaved with context-dependent reassessment. Notably, switching frequency is nearly uncorrelated with the final score ($r = -0.11$), indicating that effective adaptation depends on switching when needed rather than switching more often.

# D Method Details

## D.1 Implementation of the Shared Mode Coordinates

**Construction.** We represent the $K$ modes by a shared orthonormal frame $E = [e_1, \dots, e_K] \in \mathbb{R}^{d\times K}$. At initialization, we sample a matrix $G \in \mathbb{R}^{d\times K}$ with i.i.d. entries $G_{ij} \sim \mathcal{N}(0,1)$, compute its thin QR factorization $G = QR$, and set $E = Q$. This construction ensures $E^\top E = I_K$. The frame is stored as a non-trainable buffer and remains fixed throughout training.

**Expressivity under changes of frame orientation.** The particular orientation of the frame does not restrict the model's expressive capacity. For any two orthonormal frames $E, E' \in \mathbb{R}^{d\times K}$, there exists an orthogonal matrix $O \in \mathbb{R}^{d\times d}$ such that $E' = OE$. If a representation $z = Wh + b$ is compared with the frame, setting $W' = OW$ and $b' = Ob$ gives $z' = Oz$ and preserves its similarities to all mode coordinates:

$$(z')^\top e'_m = (Oz)^\top (Oe_m) = z^\top e_m. \tag{30}$$

The same property holds after $\ell_2$ normalization because orthogonal transformations preserve vector norms. Conversely, a linear input projection $Ae_m$ is preserved by setting $A' = AO^\top$, since

$$A'e'_m = AO^\top Oe_m = Ae_m. \tag{31}$$

Therefore, a change in frame orientation can be absorbed by the corresponding input and output projections. Gaussian QR initialization thus selects an arbitrary shared reference system without restricting the represented function class.

### D.2 Training Procedure for PRISM

Stage 1 alternates the evaluator and encoder updates in equation 10 and equation 9 while holding the mode histories fixed within each round. Stage 2 freezes the discovered modes, trains the support gate with equation 12, and then trains the Gaussian decoder with equation 15.

**Algorithm 1** Training PRISM

**Require:** Offline trajectories $\mathcal{D}$, number of modes $K$, history window $k$, fixed mode coordinates $E$

Compute $G_t$ and $y_t = \exp(G_t/\lambda)$ for all samples
Initialize $q_\theta$ and $\mu_\phi$
**Stage 1: Return-Tilted Mode Discovery**
**repeat**
    Assign mode labels from $q_\theta(m_t \mid s_t, a_t)$
    Construct $c_t$ and fix the mode histories
    Update $\phi$ by minimizing $\mathcal{L}_{\text{eval}}$ in equation 10
    Update $\theta$ by minimizing $\mathcal{L}_{\text{enc}}$ in equation 9
**until** the mode assignments converge
Freeze $q_{\theta^\star}$ and $\mu_{\phi^\star}$
**Stage 2: Mode-Gated Policy Optimization**
Set $\hat{m}_t \leftarrow \arg\max_m q_{\theta^\star}(m \mid s_t, a_t)$ for every sample
Reconstruct $c_t = (s_{t-k:t}, \hat{m}_{t-k:t-1})$
Train $g_\omega(m \mid c)$ by minimizing $\mathcal{L}_{\text{sup}}$ in equation 12
Compute $\rho(c_t)$ using equation 13
Train $\pi_\xi(a \mid s, e_m)$ by minimizing $\mathcal{L}_{\text{dec}}$ in equation 15
**return** $q_{\theta^\star}$, $\mu_{\phi^\star}$, $g_\omega$, and $\pi_\xi$

### D.3 Inference Procedure for PRISM

At deployment, PRISM first uses the support gate $g_\omega$ to prioritize modes with higher offline data support, then selects the mode $m_t^\star$ by maximizing the evaluator $\mu_{\phi^\star}$ within this subset, and decodes the action via $\pi_\xi$.

**Algorithm 2** Deployment with Support-Prioritized Mode Gating

**Require:** $\mu_{\phi^\star}$, $g_\omega$, $\pi_\xi$, support budget $k_{\text{sup}}$

Initialize the mode history
**for** $t = 0, \ldots, T-1$ **do**
    Construct $c_t = (s_{t-k:t}, m^\star_{t-k:t-1})$
    Compute $g_\omega(m \mid c_t)$ and $\mu_{\phi^\star}(c_t, m)$ for all modes
    Set $\mathcal{M}_{k_{\text{sup}}}(c_t) \leftarrow \text{Top-k}_m\, g_\omega(m \mid c_t)$
    Set $m_t^\star \leftarrow \arg\max_{m \in \mathcal{M}_{k_{\text{sup}}}(c_t)} \mu_{\phi^\star}(c_t, m)$
    Produce the action with $\pi_\xi(\cdot \mid s_t, e_{m_t^\star})$
    Execute the action and append $m_t^\star$ to the mode history
**end for**

## E Experimental Details

### E.1 Model Architecture

PRISM consists of a Mode Encoder, Mode Decoder, Support Gate, and Mode Evaluator. The encoder and decoder are implemented with MLPs, while the gate and evaluator use independently parameterized Transformer architectures. The main architectural and optimization settings are summarized in Table 4.

Table 4: Main architecture and optimization settings of PRISM. All modules use Adam.

| Module | Architecture | Layers | Hidden Dim. | Attn. Heads | Dropout | Batch Size | LR |
|---|---|---|---|---|---|---|---|
| Mode Encoder | MLP | 2 | 256 | – | 0.1 | 512 | $1 \times 10^{-4}$ |
| Mode Decoder | MLP | 2 | 256 | – | 0.1 | 256 | $5 \times 10^{-4}$ |
| Support Gate | Transformer | 4 | 256 | 4 | 0.1 | 256 | $5 \times 10^{-4}$ |
| Mode Evaluator | Transformer | 4 | 256 | 4 | 0.1 | 256 | $5 \times 10^{-4}$ |

### E.2 Hyperparameter Settings

The principal hyperparameter settings used in our experiments are summarized in Table 5. The encoder temperature $\sigma$ is annealed from $0.3$ to $0.1$ during training. For numerical stability, we normalize the return-to-go $G_t$ before computing the outcome evidence $y_t = \exp(G_t/\lambda)$. All remaining implementation details follow the default settings of the respective benchmarks.

Table 5: Main hyperparameter settings of PRISM.

| Hyperparameter | Value |
|---|---|
| Number of modes $K$ | 8 |
| Mode-frame dimension $d$ | 32 |
| Support-gate top-$k_{\text{sup}}$ | 4 |
| Temperature $\lambda$ | 1.0 |
| Discount factor $\gamma$ | 0.995 |
| Decoder weight-clipping range | $[0.01,\ 20]$ |
| Decoder log-standard-deviation range | $[-5,\ 2]$ |

### E.3 Benchmarks

#### E.3.1 AuctionNet

**Overview.** AuctionNet (Su et al., 2024) is a large-scale industrial benchmark built from real-world bidding logs, featuring highly stochastic dynamics together with strict budget and CPA constraints. Each episode corresponds to one delivery day and is divided into $48$ decision steps, with $48$ advertisers competing under distinct budgets and CPA targets. During evaluation the model controls a designated target advertiser, and the score is averaged over repeated trials across advertiser profiles and delivery periods.

**Dataset information.** The state is 16-dimensional and aggregated from the raw traffic logs: the remaining time and budget ratio, the mean bid, least winning cost, predicted conversion probability, conversion rate and win rate each summarized over the history and the last three steps, the current-step mean predicted conversion probability, and the number of ad opportunities in the current step, the last three steps and the whole history. The action is a single scalar bidding coefficient applied uniformly to all ad opportunities within a step. AuctionNet has two versions, with AuctionNet-Sparse characterized by sparse rewards; both share the same state space, action space, horizon and constraint set.

**Reward and evaluation.** Both versions maximize the number of obtained conversions under the budget and target-CPA constraints,

$$\max_{\{\alpha_i^t\}} \sum_{t=1}^{T} r_i^t \quad \text{s.t.} \quad \sum_{t=1}^{T} \left\langle \boldsymbol{x}_i^t,\, \boldsymbol{c}_i^t \right\rangle \le \omega_i, \quad \text{cpa}_i \le d_i, \tag{32}$$

where $\boldsymbol{x}_i^t$ indicates the opportunities won by agent $i$ at step $t$, $\boldsymbol{c}_i^t$ are the corresponding costs, and the per-step reward $r_i^t = \langle \boldsymbol{x}_i^t, \boldsymbol{v}_i^t \rangle$ is exactly the number of conversions obtained at step $t$, with $\boldsymbol{v}_i^t$ the value (conversion indicator) of the won opportunities. The realized cost per action is

$\mathrm{cpa}_i = \sum_t \langle \boldsymbol{x}_i^t, \boldsymbol{c}_i^t \rangle / \sum_t r_i^t$, i.e., the total cost divided by the total number of conversions. The budget constraint is enforced by the environment, whereas violations of the CPA constraint are permitted but penalized multiplicatively. Concretely, let

$$\mathrm{ratio}_i := \frac{\mathrm{cpa}_i}{d_i} \tag{33}$$

denote the ratio between the real CPA and the target CPA, and define the penalty factor

$$\mathrm{penalty}_i := \min\left\{ \frac{1}{\mathrm{ratio}_i^2},\ 1 \right\}. \tag{34}$$

The score of agent $i$ is then the total value of the won opportunities discounted by this factor,

$$\mathrm{score}_i := \left( \sum_{t=1}^{T} \sum_j o_{ij}^t\, v_{ij}^t \right) \times \mathrm{penalty}_i := \min\left\{ \left( \frac{d_i}{\mathrm{cpa}_i} \right)^2,\ 1 \right\} \sum_{t=1}^{T} r_i^t, \tag{35}$$

where $o_{ij}^t \in \{0, 1\}$ indicates whether opportunity $j$ is won at step $t$. Performance is measured by this episode score, reported under different budget ratios.

### E.3.2 D4RL GYM-MUJOCO

**Overview.** D4RL (Fu et al., 2020) is the standard benchmark for offline reinforcement learning. We use its three Gym-MuJoCo locomotion tasks—HOPPER, WALKER2D and HALFCHEETAH—in which the agent maximizes forward velocity while minimizing control costs over a horizon of 1000 steps. HALFCHEETAH never terminates early, whereas HOPPER and WALKER2D terminate upon falling, so a single out-of-distribution action can truncate an otherwise good trajectory. We use the `-v2` release, in which every trajectory is truncated at a maximum of 1000 steps and true terminations are distinguished from time-limit timeouts.

**Dataset information.** Each task provides three variants differing in the behaviour policy: *medium* (`-m`), collected by an SAC policy at roughly one third of expert performance; *medium-replay* (`-mr`), the entire replay buffer accumulated while training up to that checkpoint, mixing near-random, failing and improving behaviour; and *medium-expert* (`-me`), a 50–50 mixture of *medium* and *expert* data. The three variants thus span the data-quality spectrum from sub-optimal through mixed to high-quality.

**Reward and evaluation.** All three tasks provide a dense per-step reward trading forward progress against control effort and state-health penalties. Following standard practice, we report the normalized score

$$\mathrm{score} \;=\; 100 \times \frac{R_\pi - R_{\mathrm{random}}}{R_{\mathrm{expert}} - R_{\mathrm{random}}}, \tag{36}$$

where $R_\pi$ is the undiscounted return of the learned policy averaged over evaluation episodes and seeds, and $(R_{\mathrm{random}}, R_{\mathrm{expert}})$ is $(-20.27, 3234.3)$ for HOPPER, $(1.63, 4592.3)$ for WALKER2D and $(-280.18, 12135.0)$ for HALFCHEETAH, so that 0 and 100 correspond to random and expert-level performance. We report the per-dataset scores and their mean over the nine task–variant combinations.