Towards a Pretrained Model for Restless Bandits via Multi-arm Generalization

Yunfan Zhao[*1], **Nikhil Behari**[*1], **Edward Hughes**[2], **Edwin Zhang**[1], **Dheeraj Nagaraj**[2], **Karl Tuyls**[2], **Aparna Taneja**[2] and **Milind Tambe**[1,2]

[1]Harvard University
[2]Google

Abstract

Restless multi-arm bandits (RMABs) is a class of resource allocation problems with broad application in areas such as healthcare, online advertising, and anti-poaching. We explore several important question such as how to handle arms opting-in and opting-out over time without frequent retraining from scratch, how to deal with continuous state settings with nonlinear reward functions, which appear naturally in practical contexts. We address these questions by developing a pre-trained model (PreFeRMAB) based on a novel combination of three key ideas: (i) to enable fast generalization, we use train agents to learn from each other's experience; (ii) to accommodate streaming RMABs, we derive a new update rule for a crucial λ-network; (iii) to handle more complex continuous state settings, we design the algorithm to automatically define an abstract state based on raw observation and reward data. PreFeRMAB allows general zero-shot ability on previously unseen RMABs, and can be fine-tuned on specific instances in a more sample-efficient way than retraining from scratch. We theoretically prove the benefits of multi-arm generalization and empirically demonstrate the advantages of our approach on several challenging, real-world inspired problems.

1 Introduction

Restless multi-arm bandits (RMABs), a class of resource allocation problems involving multiple agents with a global resource constraint, have found applications in various scenarios, including resource allocation in multi-channel communication, machine maintenance, and healthcare [Hodge and Glazebrook, 2015; Mate *et al.*, 2022]. RMABs have recently been studied from a multi-agent reinforcement learning perspective.

The usual RMAB setting considers a fixed number of arms, each associated with a known, fixed MDP with finite state and action spaces; the RMAB chooses K of N arms every round to optimize some long term objective. Even in this setting, the problem has been shown to be PSPACE hard [Papadimitriou and Tsitsiklis, 1999]. Several approximation algorithms have been proposed in this setting [Whittle, 1988; Hawkins, 2003], *particularly when MDP transition probabilities are fully specified,* which are successful in practice. State-of-the-art approaches for *binary action* RMABs commonly provide policies based on the Whittle index [Whittle, 1988], an approach that has also been generalized to *multi-action* RMABs[Hawkins, 2003; Killian *et al.*, 2021b]. There are also linear programming-based approaches to both *binary and multi-action* RMABs [Zhang and Frazier, 2021]. Reinforcement learning (RL) based techniques have also been proposed as state-of-the-art solutions for general *multi-action RMABs* [Xiong and Li, 2023].

In this work, we focus on RL-based methods that provide general solutions to binary and multi-action RMABs, without requiring ground truth transition dynamics, or special properties such as indexability as required by other approaches [Wang *et al.*, 2023]. Unfortunately, several limitations exist in current RMAB solutions, especially for state of the art RL-based solutions, making them challenging or inefficient to deploy in real-world resource allocation problems.

The first limitation arises when dealing with arms that constantly opt-in (also known as streaming RMABs), which happens in public health programs where new patients (arms in RMABs) arrive asynchronously. Existing solutions either require ground truth transition probabilities [Mate *et al.*, 2021], which are often unknown in practice, or else require an entirely new model to be trained repeatedly, which can be extremely computationally costly and sample inefficient.

A second limitation occurs for new programs, or existing programs experiencing a slight change in the user base. In these situations, existing approaches do not provide a pretrained RMAB model that can be immediately deployed. In deep learning, pretrained models are the foundation for contemporary, large-scale image and text networks that generalize well across a variety of tasks [Bommasani *et al.*, 2021]. For real-world problems modeled with RMABs, establishing a similar pretrained model is essential to reduce the burden of training new RMAB policies from scratch.

The third limitation occurs in handling continuous state *multi-action* RMABs that have important applications [Sinha and Mahajan, 2022; Dusonchet and Hongler, 2003]. Naturally continuous domain state-spaces, such as patient adher-

[*]Equal contribution.

ence, are often binned into manually crafted discrete state spaces to improve model tractability and scalability [Mate *et al.*, 2022], resulting in the loss of crucial information about raw observations.

In this work we present PreFeRMAB, a **Pre**trained **F**lexibl**e** model for **RMAB**s. Using multi-arm generalization, PreFeRMAB enables *zero-shot* deployment for unseen arms as well as rapid fine-tuning for specific RMAB settings.

Our main contributions are:

- To the best of our knowledge, we are the first to develop a pretrained RMAB model with zero shot ability on entire sets of unseen arms.
- Whereas a general multiagent RL system could suffer from sample complexity exponential in the number of agents N [Gheshlaghi Azar *et al.*, 2013], we prove PreFeRMAB benefits from larger N, via multi-arm generalization and better estimation of the population distribution of arm features.
- Our pretrained model can be fine-tuned on specific instances in a more sample-efficient way than training from scratch, requiring less than 12.5% of samples needed for training a previous *multi-action* RMAB model in a healthcare setting [Verma *et al.*, 2023].
- We derive an update rule for a crucial λ-network, allowing changing numbers of arms without retraining. While streaming bandits received considerable attention [Liau *et al.*, 2018], we are the first to handle streaming *multi-action* RMABs with unknown transition dynamics.
- Our model accommodates both discrete and continuous states. To address the continuous state setting, where real-world problems often require nonlinear rewards [Riquelme *et al.*, 2018], we providing a StateShaping module to automatically define an abstract state.

2 Related Work

RMABs with binary and multiple actions. Solving an RMAB problem, even with known transition dynamics, is known to be PSPACE hard [Papadimitriou and Tsitsiklis, 1999]. For binary action RMABs, [Whittle, 1988] provides an approximate solution, using a Lagrangian relaxation to decouple arms and choose actions by computing so-called Whittle indices of each arm. It has been shown that the Whittle index policy is asymptotically optimal under the indexability condition [Weber and Weiss, 1990; Akbarzadeh and Mahajan, 2019]. The Whittle index was extended to a special class of *multi-action* RMABs with monotonic structure [Hodge and Glazebrook, 2015]. A method for more general *multi-action* RMABs based on Lagrangian relaxation was proposed by [Killian *et al.*, 2021b]. Weakly coupled Markov Decisions Processes (WCMDP), which generalizes multi-action RMABs to have multiple constraints, was studied by [Hawkins, 2003], who proposed a Langrangian decomposition approach. WCMDP was subsequently studied by [Adelman and Mersereau, 2008], who proposed improvements in solution quality at the expense of higher computational costs. While above methods developed for WCMDP require knowledge of ground truth transition dynamics, our algorithm handles unknown transition dynamics, which is more common in practice[Wang *et al.*, 2023]. Additionally, the above works in *multi-action* settings do not provide algorithms for continuous state RMABs.

Multi-agent RL and RL for RMABs. RMABs are a specific instance of the powerful multi-agent RL framework used to model systems with multiple interacting agents in both competitive and co-operative settings [Shapley, 1953; Littman, 1994], for which significant strides have been made empirically [Jaques *et al.*, 2019; Yu *et al.*, 2022] and theoretically [Jin *et al.*, 2021; Xie *et al.*, 2020]. [Nakhleh *et al.*, 2021] proposed a deep RL method to estimate the Whittle index. [Fu *et al.*, 2019] provided an algorithm to learn a Q-function based on Whittle indices, states, and actions. [Avrachenkov and Borkar, 2022] and [Biswas *et al.*, 2021] developed Whittle index-based Q-learning methods with convergence guarantees. While the aforementioned works focus on binary action RMABs, [Killian *et al.*, 2021a] generalized this to multi-action RMABs using tabular Q-learning. A subsequent work [Killian *et al.*, 2022], which focussed on robustness against adversarial distributions, took a deep RL approach that was more scalable. However, existing works on multi-action RMABs do not consider streaming RMABs and require training from scratch when a new arm opts-in. Additionally, works built on tabular Q-learning [Fu *et al.*, 2019; Avrachenkov and Borkar, 2022; Biswas *et al.*, 2021; Killian *et al.*, 2021a] may not generalize to continuous state RMABs without significant modifications. Our pretrained model addresses these limitations, and enables zero-shot ability on a wide range of unseen RMABs.

Streaming algorithms. The streaming model, pioneered by [Alon *et al.*, 1996], considers a scenario where data arrives online and the amount of memory is limited. The model is adapted to multi-arm bandits (MAB), assuming that arms arrive in a stream and the number of arms that can be stored is limited [Liau *et al.*, 2018; Chaudhuri and Kalyanakrishnan, 2020]. The streaming model has recently been adapted to *binary action* RMABs with *known* transition probabilities[Mate *et al.*, 2021], but not studied in the more general and practical settings of *multi-action* RMABs with *unknown* transition dynamics. We aim to close this gap.

Zero-shot generalization and fine-tuning. Foundation models that have a strong ability to generalize to new tasks in zero shot and efficiently adapt to new tasks via fine-tuning have received great research attention [Bommasani *et al.*, 2021]. Such models are typically trained on vast data, such as internet-scale text [Devlin *et al.*, 2018] or images [Ramesh *et al.*, 2021]. RL has seen success in the direction of foundation models for decision making, using simulated [Team *et al.*, 2023] and real-world [Yu *et al.*, 2020] environments. A pretrained model for RMABs is needed [Zhao *et al.*, 2024]. To our knowledge, we are the first to realize zero-shot generalization and efficient fine-tuning in the setting of RMABs.

3 Problem Statement

We study multi-action RMABs with system capacity N, where existing arms have the option to opt-out (that is, the state-action-rewards corresponding to them are disregarded by the model post opt-out), and new, unseen arms can request to opt-in (that is, these arms are considered only post the opt-in time). Such requests will be accepted if and only if the system capacity permits. A vector $\xi_t \in \{0,1\}^N$ represents the opt-in decisions:

$$\xi_{i,t} = \begin{cases} 1 & \text{if arm } i \text{ opts-in at round } t, \\ 0 & \text{otherwise.} \end{cases}$$

Notice that existing arms must opt-in in each round t to remain in the system. For each arm $i \in [N]$, the state space $\mathcal{S}_i$ can be either discrete or continuous, and the action space $\mathcal{A}_i$ is a finite set of discrete actions. Each action $a \in \mathcal{A}_i$ has an associated cost $\mathcal{C}_i(a)$, with $\mathcal{C}_i(0)$ denoting a no-cost passive action. The reward at a state is given by a function $R_i : \mathcal{S}_i \to \mathbb{R}$. We let $\beta \in [0,1)$ denote a discount factor. Each arm has a unique feature vector $\boldsymbol{z}_i \in \mathbb{R}^m$ that provides useful information about the arm. Notice our model directly utilizes feature information in its policy network, without requiring intermediate steps to extract transition dynamics information from features.

When the state space is discrete, each arm $i \in [N]$ follows a Markov Decision Process $(\mathcal{S}_i, \mathcal{A}_i, \mathcal{C}_i, T_i, R_i, \beta, \boldsymbol{z}_i)$, where $T_i : \mathcal{S}_i \times \mathcal{A}_i \times \mathcal{S}_i \to [0,1]$ is a transition matrix representing the probability of transitioning from the current state to the next state given an action. In contrast, when the state space is continuous, each arm $i \in [N]$ follows a Markov Decision Process $(\mathcal{S}_i, \mathcal{A}_i, \mathcal{C}_i, \boldsymbol{\Gamma}_i, R_i, \beta, \boldsymbol{z}_i)$, where $\boldsymbol{\Gamma}_i$ is a set of parameters encoding the transition dynamics. For example, in the case that the next state moves according to a Gaussian distribution, $\boldsymbol{\Gamma}_i$ may denote the mean and variance of the Gaussian.

For simplicity, we assume that $\mathcal{S}_i, \mathcal{A}_i, \mathcal{C}_i$, and R_i are the same for all arms $i \in [N]$ and omit the subscript i. Note that our algorithms can also be used in the general case where rewards and action costs are different across arms. For ease of notation, we let $\boldsymbol{s} \in \mathbb{R}^N$ denote the state over all arms, and we let $\boldsymbol{A} \in \{0,1\}^{N \times |\mathcal{A}|}$ denote one-hot-encoding of the actions taken over all arms. The agent learns a policy π that maps states $\boldsymbol{s}$ and features $\boldsymbol{z}$ to actions $\boldsymbol{A}$, while satisfying a constraint that the sum cost of actions taken is no greater than a given budget B in every timestep $t \in [H]$, where H is the length of the horizon.

Our goal is to learn an RMAB policy that maximizes the following Bellman equation The key difficulty in learning such a policy is how to utilize features $\boldsymbol{z}$ and address opt-in decisions $\boldsymbol{\xi}$. These are important research questions not addressed in previous works [Killian *et al.*, 2022].

$$J(\boldsymbol{s}, \boldsymbol{z}, \boldsymbol{\xi}) = \max_{\boldsymbol{A}} \left\{ \sum_{i=1}^{N} R(\boldsymbol{s}_i) + \beta\, \mathbb{E}\left[J(\boldsymbol{s}', \boldsymbol{z}, \boldsymbol{\xi}) \mid \boldsymbol{s}, \boldsymbol{A} \right] \right\}, \tag{1}$$

$$\text{s.t.} \sum_{i=1}^{N} \sum_{j=1}^{|\mathcal{A}|} \boldsymbol{A}_{ij} c_j \le B \quad \text{and} \quad \sum_{j=1}^{|\mathcal{A}|} \boldsymbol{A}_{ij} = 1 \quad \forall i \in [N],$$

where $c_j \in \mathcal{C}$ is the cost of j^{th} action, and $A_{ij} = 1$ if action j is chosen on arm i and $A_{ij} = 0$ otherwise. Further, we assume that the rewards R are uniformly bounded by $R_{\max}$.

4 Generalized Model for RMABs

We first provide an overview of key ideas and then discuss each of the ideas in more detail. (See Figure 3 in Appendix for an overview of the training procedure.)

4.1 Key Algorithmic Ideas

Several key algorithmic novelties are necessary for our model to address limitations of existing works:

A pretrained model via multi-arm generalization: We train agents to learn from each others' experience. Whereas a general multiagent RL system could suffer from sample complexity exponential in the number of arms N [Gheshlaghi Azar *et al.*, 2013], we prove PreFeRMAB benefits from a larger N, via generalization across arms.

A novel λ-network updating rule for opt-in: The opt-in and opt-out of arms induce a more complex form of the Lagrangian and add randomness to actions taken by agents. We provide a new λ-network update rule and train PreFeRMAB with opt-in and opt-out of arms, to enable zero-shot performance across various opt-in rates and accommodate streaming RMABs.

Handling continuous states with StateShaping subroutine: In the continuous state setting, real-world problems often require nonlinear rewards [Riquelme *et al.*, 2018], and naively using raw observations to train models may result in poor performance (see Figure 5). To tackle this challenge, we design the algorithm to automatically define an abstract state based on raw observation and reward data.

4.2 A Pretrained Model via Multi-arm Generalization

To enable multi-arm generalization, we introduce feature-based Q-values, together with a Lagrangian relaxation with features $\boldsymbol{z}_i$ and opt-in decisions ξ_i:

$$\begin{aligned} & J(s, \boldsymbol{z}, \boldsymbol{\xi}, \lambda^\star) \\ &= \min_{\lambda \ge 0} \left(\frac{\lambda B}{1-\beta} + \sum_{i=1}^{N} \max_{a_i \in |\mathcal{A}|} \left\{ Q(\boldsymbol{s}_i, a_i, \boldsymbol{z}_i, \xi_i, \lambda) \right\} \right), \quad (2) \\ & \text{s.t. } Q(\boldsymbol{s}_i, a_i, \boldsymbol{z}_i, \xi_i, \lambda) \\ &= \xi_i R(\boldsymbol{s}_i) - \xi_i \lambda c_{a_i} + \beta\, \mathbb{E}\left[Q(\boldsymbol{s}'_i, a_i, \boldsymbol{z}_i, \xi_i, \lambda) \mid \pi(\lambda) \right]. \end{aligned}$$

where Q is the Q-function, a_i is the action of arm i, s'_i is the state transitioned to from s_i under action a_i, and $\pi(\lambda)$ is the optimal policy under a given λ. Notice that this relaxation decouples the Q-functions of the arms, and therefore Q_i can be solved independently for a given λ.

Now we discuss how we use feature-based Q-values and how agents could learn from each other. During pretraining, having received arms' opt-in and out decisions (line 5), Algorithm 1 samples an action-charge λ based on updated opt-in decisions ξ and features $\boldsymbol{z}_i$ (line 6). Next, from opt-in arms we collect trajectories (lines 7-14), which are later used to train a single pair of actor/critic networks for all arms, allowing the policy for one arm to benefit from other arms' data. After that, we update the policy network θ and the critic network ϕ (Line 16), using feature-based Q-values to compute advantage estimates for the actor in PPO update. Critically, feature-based Q-values updated with one arm's data, improves the policy for other arms. In real-world problems with missing feature entries or less informative features, it is more important for agents to learn from each other (see Table 1 in Sec 5.2). Intuitively, if a model only learns from homogeneous arms, then we should expect this model to perform poorly when used out-of-the-box on arms with completely different behaviors.

Algorithm 1 PreFeRMAB (Training)

Input: n_epochs, n_steps, λ-update frequency $K \in \mathbb{N}^+$, and system capacity N
Initialize actor θ, critic ϕ, λ-network Λ, buffer = [], state $\boldsymbol{s} \in \mathbb{R}^N$, and features $\boldsymbol{z}_i \in \mathbb{R}^m$
Initialize *StateShaping*, and set $\bar{\boldsymbol{s}} \leftarrow$*StateShaping*$(\boldsymbol{s})$
for epoch = 1, 2, . . . , n_epochs **do**
  Receive opt-in/out requests and update ξ and $\boldsymbol{z}_i$.
  Compute $\lambda = \Lambda(\bar{\boldsymbol{s}}, \{z_i\}_{i=1}^N, \boldsymbol{\xi})$
  for timestep t = 1, . . . , n_steps **do**
    for Arm i = 1, . . . , N **do**
      if Arm i is opt-in (i.e. $\xi_i = 1$) **then**
        Sample an action $a_i \sim \theta(\bar{s}_i, \lambda, \boldsymbol{z}_i)$
        $s_i', r_i = \text{Simulate}(s_i, a_i)$
        $\bar{s_i'} =$ *StateShaping*(s_i')
        Add tuple $(s_i, \bar{s_i}, a_i, r_i, \bar{s_i'}, \boldsymbol{z}_i)$ to buffer
        $s_i \leftarrow s_i'$, $\bar{s}_i \leftarrow \bar{s_i'}$
  Add tuple $(\lambda, \boldsymbol{\xi})$ to buffer
  Update the (θ, ϕ) pair using buffer.
  if epoch // K = 0 **then**
    Update Λ via Prop 2 using trajectories in buffer
    Update $\hat{r}(\cdot)$ in *StateShaping* using (s, r)-tuples in buffer

We will now give a theoretical guarantee of multi-arm generalization by considering the following simplified setting and assumptions where we do not consider opt-in and opt-out. That is, in each epoch we draw a new set of N arms. We let the distribution of arm features (i.e, $\mathbf{z}_i$) to be denoted by the probability measure μ^*. Each 'sample' for our policy network training consists of N features corresponding to N arms $(\mathbf{z}_1, \ldots, \mathbf{z}_N)$, drawn i.i.d. from the distribution μ^*. Call the empirical distribution of $(\mathbf{z}_i)$ to be $\hat{\mu}$. During training, we receive n_{epochs} i.i.d. draws of N arm features each, denoted by $\hat{\mu}_1, \ldots, \hat{\mu}_{n_{\text{epochs}}}$

Let Θ denote the space of neural network weights of the policy network (for clarity, we shorten the (θ, ϕ) in Algorithm 1 to θ). The neural network inputs are Lagrangian multiplier λ, state of an arm s, its feature $\mathbf{z}$ and the output is $a \in \mathcal{A}$. Let $V(\mathbf{s}, \theta, \lambda, \hat{\mu})$ denote the discounted reward, averaged over N arms with features $\hat{\mu}$ obtained with the neural network with parameter θ, starting from the state $\mathbf{s}$ (cumulative state of all arms). The proposition below shows the generalization properties of the output of Algorithm 1. The proof and a detailed discussion of the assumptions and consequences are given in Section D.

Proposition 1. *Suppose the following assumptions hold:*

1. *Algorithm 1 learns neural network weights $\hat{\theta} \in \Theta$, whose policy is optimal for each $(\hat{\mu}_i, \lambda)$ for $1 \leq i \leq n_{\text{epochs}}$ and $\lambda \in [0, \lambda_{\max}]$*
2. *There exists $\theta^* \in \Theta$ which is optimal for every instance $(\hat{\mu}, \lambda)$.*
3. *$\Theta = \mathcal{B}_2(D, \mathbb{R}^d)$, the ℓ_2 ball of radius D in $\mathbb{R}^d$.*
4. *$|V(\mathbf{s}, \theta_1, \lambda, \hat{\mu}) - V(\mathbf{s}, \theta_2, \lambda, \hat{\mu})| \leq L\|\theta_1 - \theta_2\|$ and $|V(\mathbf{s}, \theta, \lambda_1, \hat{\mu}) - V(\mathbf{s}, \theta, \lambda_2, \hat{\mu})| \leq L|\lambda_1 - \lambda_2|$ for all $\theta_1, \theta_2, \theta \in \Theta$ and $\lambda_1, \lambda_2, \lambda \in [0, \lambda_{\max}]$.*

Then, the generalization error over unseen arms ($\hat{\mu}$) satisfies:

$$\mathbb{E}_{\hat{\mu},\hat{\theta}}\big[\inf_{\lambda \in [0,\lambda_{\max}]} V(\mathbf{s}, \hat{\theta}, \lambda, \hat{\mu})\big] \geq \mathbb{E}_{\hat{\mu}}\big[\inf_{\lambda \in [0,\lambda_{\max}]} V(\mathbf{s}, \theta^*, \lambda, \hat{\mu})\big] - \tilde{O}\left(\frac{1}{\sqrt{n_{\text{epochs}} N}}\right) \tag{3}$$

Here, $\tilde{O}$ hides polylogarithmic factors in n_{epochs}, N and constants depending on $d, D, L, \beta, \frac{B}{N}, c_j, R_{\max}$ and $\lambda_{\max}$

The assumption of existence of θ^* is reasonable: This means that there exists a neural network which gives the optimal policy for a family of single-arm MDPs indexed by $(\mathbf{z}, \lambda)$. Proposition 1 shows that when n_{epochs} and N are large, the Lagrangian relaxed value function of the learned network is close to that of the optimal network.

An important insight is that the generalization ability of the PreFeRMAB network becomes better as the number of arms per instance becomes larger. This is counter intuitive since a system with a larger number of agents are generally very complex. Jointly, the arms form an MDP with $|\mathcal{S}|^N$ states and $|\mathcal{A}|^N$ actions. General multi-agent RL problems with N arms thus can suffer from an exponential dependence on N in their sample complexity for learning (see sample complexity lower bounds in [Gheshlaghi Azar *et al.*, 2013]). However, due to the structure of RMABs and the Lagrangian relaxation, we achieve a better generalization with a larger N. Our proof in the appendix shows that this is due to the fact that a larger number of arms helps estimate the population distribution μ^* of the arm features better. We show in Table 1 that indeed having more number of arms helps the PreFeRMAB network generalize better over unseen instances.

4.3 A Novel λ-network Updating Rule

In real-world health programs, we may observe new patients constantly opt-in [Mate *et al.*, 2021]. The opt-in / opt-out decisions render the updating rule in [Killian *et al.*, 2022] unusable and add additional randomness to actions taken by the agent. To overcome this challenge and to stabilize training, we develop a new λ-network updating rule.

Proposition 2. *[λ-network updating rule] The equation for gradient descent for the objective (Eq 2) with respect to λ, with step size α is:*

$$\begin{aligned}\Lambda_t =& \Lambda_{t-1} - \alpha\left(\frac{B}{1-\beta}\right) \\ &- \alpha\left(\sum_{i=1}^{N} \mathbb{E}\left[\sum_{t=0}^{H} \xi_{i,t}\beta^t c_{i,t} + (1-\xi_{i,t})\beta^t c_{0,t}\right]\right),\end{aligned}$$

where $c_{i,t}$ is the cost of the action taken by the optimal policy on arm i in round t.

Critically, this update rule allows PreFeRMAB to handle streaming RMABs, accommodating a changing number of arms without retraining and achieving strong zero-shot performance across various opt-in rates (see Table 3 and 14). Having established an updating rule, we provide a convergence guarantee. The proofs are relegated to Appendix E.

Proposition 3 (Convergence of λ-network)**.** *Suppose the arm policies converge to the optimal Q-function for a given* Λ_t*, then the update rule (in Prop 2) for the* λ*-network converges to the optimal as the number of training epochs and the number of actions collected in each epoch go to infinity.*

4.4 Handling Continuous States with StateShaping

Real-world problems may require continuous states with nonlinear rewards [Riquelme *et al.*, 2018]. Existing RMAB algorithms either use a human-crafted discretization or fail to address challenging nonlinear rewards [Killian *et al.*, 2022]. Discretization may result in loss of information and fail to generalize to different population sizes. For example, the popular SIS epidemic model [Yaesoubi and Cohen, 2011] is expected to scale to a continuum limit as the population size increases to infinity, and a continuous state-space model can better handle scaling by using proportions instead of absolute numbers. Under nonlinear rewards, naively using raw observations in training may result in poor performance (see Figure 5). We provide a StateShaping module to improve model stability and performance.

Algorithm 2 StateShaping Subroutine

1: Input: estimator $\in$ {Isotonic Regression, KNN}, states $\boldsymbol{s} \in \mathbb{R}^N$, data $\mathcal{D}$ of (s, r) tuples
2: Output $\bar{\boldsymbol{s}} = \boldsymbol{s}$ if no normalization is desired
3: Compute
$$r_{min} = \min_{s'\colon s' \in \mathcal{D}} r(s'), \quad r_{max} = \max_{s'\colon s' \in \mathcal{D}} r(s')$$
$$s_{min} = \min_{s' \in \mathcal{D}} s', \quad s_{max} = \max_{s' \in \mathcal{D}} s'$$
4: Compute $\hat{r}(s_i)$ using the choice of Estimator.
5: Output $\bar{\boldsymbol{s}}$, where $\bar{s}_i = \frac{\hat{r}(s_i) - r_{\min}}{r_{\max} - r_{\min}}(s_{\max} - s_{\min}), \forall i$

In Algorithm 2, users can choose whether to obtain abstract state [Abel *et al.*, 2018] from raw observations (lines 2). We compute ranges of reward and raw observations, and obtain an reward estimate (lines 3-4). After that, we automatically refine the raw observation such that reward is a linear function of the abstract state (line 4), improving model stability for challenging reward functions. Here a key assumption is that reward is an increasing function of the raw observation, which is common in RMABs [Killian *et al.*, 2022]. Notice as we collect more observations, the accuracy of the reward estimator $\hat{r}(\cdot)$ will improve (it is updated in line 24 of Algorithm 1).

StateShaping instantiates the idea of state abstraction, which is shown to improve generalizability and robustness [Li *et al.*, 2006], in the RMAB context for continuous states. Applying Theorem 3 in [Li *et al.*, 2006] to the Lagrangian relaxation (Eq. 2), we have that an optimal policy learnt using the abstract state space is guaranteed to be also optimal in the ground MDP (defined by raw observations).

4.5 Inference using Pretrained Model

An important difference between training and inference is that during inference time, we strictly enforce the budget constraint on the trained model, by greedily selecting highest probability actions until the budget is reached. The rest of the inference components are similar to the training component.

Algorithm 3 PreFeRMAB (Inference)

1: Input: : States $\boldsymbol{s}$, costs C, budget B, features $\boldsymbol{z}_i \in \mathbb{R}^m$, opt-in decisions $\boldsymbol{\xi}$, agent actor θ, λ-network, *StateShaping* routine with trained estimator $\hat{r}(\cdot)$
2: Compute $\lambda = \Lambda(\bar{\boldsymbol{s}}, \{z_i\}_{i=1}^N, \boldsymbol{\xi})$
3: **for** Arm i = 1, . . . , N **do**
4:     **if** Arm i is opt-in (i.e. $\xi_i = 1$) **then**
5:         $\bar{s_i} = \mathit{StateShaping}(s_i)$
6:         Compute $p_i \sim \theta(\bar{s}_i, \lambda, \boldsymbol{z}_i)$
7: $\boldsymbol{a}$ = GreedyProba(p, C, B) ▷ Greedily select highest probability actions until budget B is reached

5 Experimental Evaluation

We provide experimental evaluations of our model in *three separate domains*, including a synthetic setting, an epidemic modeling setting, as well as a maternal healthcare intervention setting. We first describe these three experimental domains. Then, we provide results for PreFeRMAB in a *zero-shot evaluation setting*, demonstrating the performance of our model on *new, unseen test arms* drawn from distributions distinct from those in training. Here, we demonstrate the flexibility of PreFeRMAB, including strong performance across domains, state representations (discrete vs. continuous), and over various challenging reward functions. Finally, we demonstrate the strength of using PreFeRMAB as a *pretrained model*, enabling faster convergence for fine-tuning on a specific set of evaluation arms.

In Appendix B, we provide **ablation studies** over (1) a wider range of opt-in rates (2) different feature mappings (3) DDLPO topline with and without features (4) more problem settings. For hyperparameter details, we refer to Appendix A.

5.1 Experimental Settings

Features: In all experiments, we generate features by projecting parameters that describe the ground truth transition dynamics into features using randomly generated projection matrices. The dimension of feature equals the number of parameters required to describe the transition dynamics. In Appendix B, we provide results on different feature mappings.

Synthetic: Following [Killian *et al.*, 2022], we consider a synthetic dataset with binary states and binary actions. The transition probabilities for each arm i are represented by matrices $T^{(i)}_{s=0}$ and $T^{(i)}_{s=1}$ for arm i at states 0 and 1 respectively:

$$T^{(i)}_{s=0} = \begin{bmatrix} p_{00} & 1 - p_{00} \\ p_{01} & 1 - p_{01} \end{bmatrix}, \quad T^{(i)}_{s=1} = \begin{bmatrix} p_{10} & 1 - p_{10} \\ p_{11} & 1 - p_{11} \end{bmatrix}$$

Each p_{jk} corresponds to the probability of transitioning from state j to state 0 when action k is taken. These values are sampled uniformly from the intervals:

$$p_{00} \in [0.4, 0.6], p_{01} \in [0.4, 0.6], p_{10} \in [0.8, 1], p_{11} \in [0.0, 1]$$

SIS Epidemic Model: Inspired by the vast literature on agent-based epidemic modeling, we adapt the SIS model given in [Yaesoubi and Cohen, 2011], following a similar experiment setup as described in [Killian *et al.*, 2022]. Arms p represent a subpopulation in distinct geographic regions; states s are the number of uninfected people within each arm's total population N_p; the number of possible states is S. Transmission within each arm is guided by parameters: κ, the average number of contacts within the arm's subpopulation in each round, and $r_{\textit{infect}}$, the probability of becoming infected after contact with an infected person.

In this setting, there is a budget constraint over interventions. There are three available intervention actions a_0, a_1, a_2 that affect the transmission parameters: a_0 represents no action; a_1 represents messaging about physical distancing; a_2 represents the distribution of face masks. We discuss additional details in Appendix A.

ARMMAN: Similar to the set up in [Biswas *et al.*, 2021; Killian *et al.*, 2022], we model the real world maternal health problem as a discrete state RMAB. We aim to encourage engagement with automated health information messaging. There are three possible states, presenting self-motivated, persuadable, and lost cause. The actions are binary. There are 6 uncertain parameters per arm, sampled from uncertainty intervals of 0.5 centered around the transition parameters that align with summary statistics given in [Biswas *et al.*, 2021].

Continuous State Modeling: Continuous state restless bandits have important applications [Lefévre, 1981; Sinha and Mahajan, 2022; Dusonchet and Hongler, 2003]. By not explicitly having a switch in the model (switching between discrete and continuous state space), we enable greater model flexibility. To demonstrate this, we consider both a Continuous Synthetic and a Continuous SIS modeling setting. We provide details of these settings in Appendix A.3.

We present **additional details**, including hyperparameters and StateShaping illustration in Appendix A.

5.2 PreFeRMAB Zero-Shot Learning

We first consider three challenging datasets in the discrete state space. After that, we present results on datasets with continuous state spaces with more complex reward functions and transition dynamics.

Pretraining. For each pretraining iteration, we sample from a binomial with mean 0.8 to determine which arms will be opted-in given system capacity N. For new arms, we sample new transition dynamics to allow the model to see a wider range of arm features.

Evaluation. We compare PreFeRMAB to *Random Action* and *No Action* baselines. In every table in this subsection, we present the *reward per arm* averaged over 50 trials, on *new, unseen arms* arm sampled from the testing distribution.

Multi-arm Generalization: Table 1 on Synthetic illustrates that PreFeRMAB, learning from *multi-arm generalization*, achieves stronger performance when the number of unique arms (i.e. arms with unique features) seen during pretraining increases. Additionally, in practice arm features may be missing or not always reliable, such as in real-world ARMMAN data [Mate *et al.*, 2022]. Our results demonstrate that when features are masked, arms could learn from similar arms' experience.

System capacity $N = 21$. Budget $B = 7$.			
# Unique training arms	45	33	21
No Action	$2.88_{\pm 0.17}$	$2.88_{\pm 0.17}$	$2.88_{\pm 0.17}$
Random Action	$3.25_{\pm 0.22}$	$3.25_{\pm 0.22}$	$3.25_{\pm 0.22}$
PreFeRMAB (2/4 Feats. Masked)	$3.81_{\pm 0.23}$	$3.79_{\pm 0.22}$	$3.59_{\pm 0.21}$
PreFeRMAB (1/4 Feats. Masked)	$3.92_{\pm 0.24}$	$3.70_{\pm 0.21}$	$3.58_{\pm 0.20}$
PreFeRMAB (0/4 Feats. Masked)	$4.02_{\pm 0.26}$	$3.80_{\pm 0.22}$	$3.78_{\pm 0.21}$

Table 1: Multi-arm generalization results on Synthetic (opt-in 100%). With the same total amount of data, PreFeRMAB achieves stronger performance when pretrained on more unique arms, especially when input arm features are masked.

Wasserstein Distance	**0.05**	**0.10**	**0.15**	**0.20**	**0.25**
System capacity $N = 48$. Budget $B = 16$.					
No Action	$3.07_{\pm 0.10}$	$2.89_{\pm 0.08}$	$2.68_{\pm 0.07}$	$2.49_{\pm 0.09}$	$2.35_{\pm 0.07}$
Random Action	$3.49_{\pm 0.09}$	$3.25_{\pm 0.09}$	$2.99_{\pm 0.16}$	$2.80_{\pm 0.17}$	$2.57_{\pm 0.17}$
PreFeRMAB	$4.50_{\pm 0.09}$	$4.30_{\pm 0.10}$	$3.81_{\pm 0.17}$	$3.79_{\pm 0.18}$	$3.46_{\pm 0.12}$
System capacity $N = 96$. Budget $B = 32$.					
No Action	$3.09_{\pm 0.08}$	$2.88_{\pm 0.04}$	$2.74_{\pm 0.05}$	$2.62_{\pm 0.06}$	$2.49_{\pm 0.06}$
Random Action	$3.44_{\pm 0.14}$	$3.23_{\pm 0.09}$	$3.05_{\pm 0.09}$	$2.90_{\pm 0.10}$	$2.70_{\pm 0.11}$
PreFeRMAB	$4.44_{\pm 0.13}$	$4.26_{\pm 0.13}$	$4.12_{\pm 0.13}$	$3.97_{\pm 0.16}$	$3.75_{\pm 0.12}$

Table 2: Results on Synthetic (opt-in 100%). For each system capacity, we pretrain a model and present zero-shot results under different amounts of distributional shift.

Number of arms / System capacity	**80%**	**85%**	**90%**	**95%**	**100%**
Parameters a_1^{eff}, a_1^{eff} are uniformly sampled from $[2, 8]$.					
No Action	$5.23_{\pm 0.17}$	$5.27_{\pm 0.16}$	$5.28_{\pm 0.16}$	$5.26_{\pm 0.14}$	$5.28_{\pm 0.13}$
Random Action	$6.94_{\pm 0.15}$	$7.00_{\pm 0.16}$	$7.03_{\pm 0.15}$	$6.97_{\pm 0.14}$	$6.99_{\pm 0.12}$
PreFeRMAB	$7.64_{\pm 0.27}$	$7.75_{\pm 0.25}$	$7.96_{\pm 0.18}$	$7.80_{\pm 0.16}$	$7.82_{\pm 0.11}$
Parameters a_1^{eff}, a_1^{eff} are uniformly sampled from $[3, 9]$.					
No Action	$5.29_{\pm 0.16}$	$5.30_{\pm 0.17}$	$5.29_{\pm 0.15}$	$5.26_{\pm 0.14}$	$5.28_{\pm 0.13}$
Random Action	$7.21_{\pm 0.15}$	$7.28_{\pm 0.18}$	$7.26_{\pm 0.15}$	$7.22_{\pm 0.13}$	$7.22_{\pm 0.12}$
PreFeRMAB	$7.77_{\pm 0.29}$	$7.87_{\pm 0.28}$	$7.90_{\pm 0.22}$	$7.95_{\pm 0.16}$	$7.95_{\pm 0.11}$

Table 3: Results on SIS ($N = 20, B = 16, S = 150$). We pretrain a model and present zero-shot results on various distributions. During training, a_1^{eff}, a_1^{eff} are uniformly sampled from $[1, 7]$.

Discrete State Settings with Different Distributional Shifts: Results on Synthetic (Table 2) shows PreFeRMAB consistently outperforms under varying amounts of distributional shift, measured in Wasserstein distance. Results on SIS (Table 3) shows PreFeRMAB performs well in settings with large state space $S = 150$ and multiple actions, under various testing distributions and opt-in rates. Results on ARMMAN (Table 4) shows PreFeRMAB could handle more *challenging* settings that mimics the scenario of a real-world non-profit organization using RMABs to allocate resources.

Continuous State Settings: Our results (Figure 1) show

Number of arms / System capacity	80%	85%	90%	95%	100%
40% motivated, 20% persuadable, and 40% lost cause.					
No Action	$2.39_{\pm 0.30}$	$2.32_{\pm 0.28}$	$2.16_{\pm 0.25}$	$2.26_{\pm 0.28}$	$2.25_{\pm 0.30}$
Random Action	$3.04_{\pm 0.40}$	$3.06_{\pm 0.38}$	$3.00_{\pm 0.36}$	$3.14_{\pm 0.36}$	$3.24_{\pm 0.32}$
PreFeRMAB	$5.47_{\pm 0.41}$	$5.00_{\pm 0.37}$	$4.95_{\pm 0.29}$	$5.34_{\pm 0.27}$	$5.03_{\pm 0.37}$
40% motivated, 40% persuadable, and 20% lost cause.					
No Action	$2.07_{\pm 0.29}$	$2.19_{\pm 0.28}$	$2.19_{\pm 0.30}$	$2.05_{\pm 0.24}$	$2.17_{\pm 0.28}$
Random Action	$3.04_{\pm 0.37}$	$3.02_{\pm 0.33}$	$2.99_{\pm 0.30}$	$3.12_{\pm 0.31}$	$3.15_{\pm 0.29}$
PreFeRMAB	$5.06_{\pm 0.36}$	$4.81_{\pm 0.35}$	$5.13_{\pm 0.34}$	$5.01_{\pm 0.26}$	$5.00_{\pm 0.28}$

Table 4: Results on ARMMAN ($N = 25, B = 7, S = 3$). We pretrain a model and present zero-shot results on various testing distributions. During training, the proportion of self-motivated, persuadable, and lost cause arms are 20%, 20%, annd 60% respectively.

that StateShaping is crucial in handling continuous states, where the reward function can be more challenging. We provide additional evaluations in Table 5, showing PreFeRMAB outperforms in complex transition dynamics. More details are provided in Appendix A.

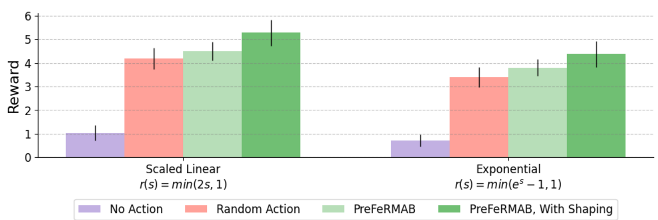

Figure 1: Results for *Continuous Synthetic* domain (N=21,B=7.0) *with challenging rewards* $r(s)$.

Number of arms / System capacity	80%	85%	90%	95%	100%
Continuous Synthetic (N=21, B=7.0, S=2)					
No Action	$0.70_{\pm 0.25}$	$0.71_{\pm 0.25}$	$0.70_{\pm 0.23}$	$0.70_{\pm 0.23}$	$0.66_{\pm 0.21}$
Random Action	$3.44_{\pm 0.48}$	$3.43_{\pm 0.45}$	$3.45_{\pm 0.41}$	$3.37_{\pm 0.41}$	$3.20_{\pm 0.38}$
PreFeRMAB	$3.94_{\pm 0.31}$	$3.76_{\pm 0.33}$	$4.01_{\pm 0.31}$	$4.02_{\pm 0.29}$	$3.67_{\pm 0.27}$
Continuous SIS Model (N=20, B=16)					
No Action	$5.64_{\pm 0.25}$	$5.68_{\pm 0.19}$	$5.57_{\pm 0.21}$	$5.48_{\pm 0.18}$	$5.62_{\pm 0.17}$
Random Action	$7.11_{\pm 0.22}$	$7.31_{\pm 0.22}$	$7.23_{\pm 0.22}$	$7.24_{\pm 0.21}$	$7.18_{\pm 0.17}$
PreFeRMAB	$7.91_{\pm 0.17}$	$8.08_{\pm 0.11}$	$7.95_{\pm 0.14}$	$7.98_{\pm 0.13}$	$7.82_{\pm 0.12}$

Table 5: Results on continuous states. For each problem instance, we pretrain a model.

Comparison with an Additional Baseline: DDLPO [Killian *et al.*, 2022] could not handle distributional shifts or various opt-in rates, the more challenging settings that PreFeRMAB is designed for. Nevertheless, we provide comparisons with DDLPO in settings with no distributional shift (Table 6, see also Appendix B.4). Notably, PreFeRMAB zero-shot performance on *unseen* arms is near that of DDLPO, which is trained and tested on the same set of arms.

5.3 PreFeRMAB Fast Fine-Tuning

Having shown the zero-shot results of PreFeRMAB, we now demonstrate finetuning capabilities of the pretrained model. In Figure 2, we compare the number of samples required to *train DDLPO from scratch* vs. the number of samples for *fine-tuning PreFeRMAB* starting from a pre-trained model (additional results in Appendix A.5). Results suggests the cost of pretraining can be amortized over different downstream instances. A non-profit organization using RMAB models may have new beneficiaries opting in every week, and training a new model from scratch every week can be 3-20 times more expensive than fine-tuning our pretrained model.

Number of arms / System capacity	80%	85%	90%	95%	100%
Synthetic with $N = 96, B = 32, S = 2$.					
No Action	$3.22_{\pm 0.12}$	$3.24_{\pm 0.12}$	$3.19_{\pm 0.11}$	$3.18_{\pm 0.11}$	$3.18_{\pm 0.11}$
Random Action	$3.62_{\pm 0.13}$	$3.66_{\pm 0.13}$	$3.58_{\pm 0.13}$	$3.60_{\pm 0.12}$	$3.60_{\pm 0.12}$
PreFeRMAB	$4.63_{\pm 0.12}$	$4.71_{\pm 0.12}$	$4.53_{\pm 0.13}$	$4.47_{\pm 0.12}$	$4.61_{\pm 0.10}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$4.58_{\pm 0.13}$
SIS with $N = 20, B = 16, S = 150$.					
No Action	$5.33_{\pm 0.16}$	$5.30_{\pm 0.15}$	$5.31_{\pm 0.14}$	$5.29_{\pm 0.13}$	$5.28_{\pm 0.13}$
Random Action	$7.03_{\pm 0.17}$	$7.13_{\pm 0.16}$	$7.02_{\pm 0.14}$	$7.11_{\pm 0.13}$	$7.06_{\pm 0.13}$
PreFeRMAB	$8.35_{\pm 0.12}$	$8.38_{\pm 0.11}$	$8.26_{\pm 0.11}$	$8.10_{\pm 0.11}$	$8.00_{\pm 0.10}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$8.09_{\pm 0.11}$
ARMMAN with $N = 25, B = 7, S = 3$.					
No Action	$2.12_{\pm 0.26}$	$2.30_{\pm 0.29}$	$2.29_{\pm 0.27}$	$2.19_{\pm 0.23}$	$2.26_{\pm 0.25}$
Random Action	$2.86_{\pm 0.32}$	$3.27_{\pm 0.40}$	$3.01_{\pm 0.30}$	$3.09_{\pm 0.35}$	$2.96_{\pm 0.31}$
PreFeRMAB	$5.06_{\pm 0.34}$	$5.26_{\pm 0.33}$	$4.68_{\pm 0.33}$	$4.75_{\pm 0.35}$	$4.61_{\pm 0.27}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$4.68_{\pm 0.09}$

Table 6: Comparison of PreFeRMAB zero-shot performance on unseen arms against that of DDLPO trained and tested on the same set of arms. For each problem instance, we pretrain a model.

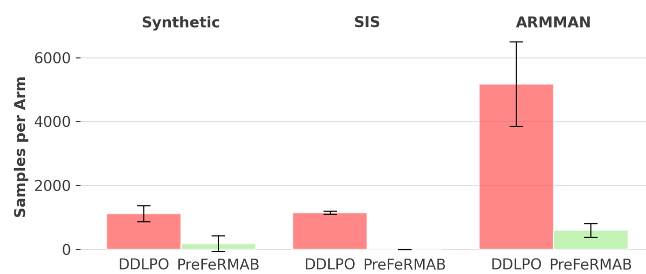

Figure 2: Comparison of samples per arm required by DDLPO and PreFeRMAB (fine-tuning using a pretrained model) to achieve maximum DDLPO reward across different environments. PreFeRMAB achieves the maximum topline reward with significantly fewer samples than DDLPO. Averages across training seeds are reported as interquartile means.

6 Conclusion

Our pretrained model (PreFeRMAB) leverages multi-arm generalization, a novel update rule for a crucial λ-network, and a StateShaping module for challenging reward functions. PreFeRMAB demonstrates general zero-shot ability on unseen arms, and can be fine-tuned on specific instances in a more sample-efficient way than training from scratch.

Acknowledgments

The work is supported by Harvard HDSI funding.

Ethical Statement

The presented methods do not carry direct negative societal implications. However, training reinforcement learning models should be done responsibly, especially given the safety concerns associated with agents engaging in extreme, unsafe, or uninformed exploration strategies. While the domains we considered such as ARMMAN do not have these concerns, the approach may be extended to extreme environments; in these cases, ensuring a robust approach to training reinforcement models is critical.

References

[Abel *et al.*, 2018] David Abel, Dilip Arumugam, Lucas Lehnert, and Michael Littman. State abstractions for lifelong reinforcement learning. In *International Conference on Machine Learning*, pages 10–19. PMLR, 2018.

[Adelman and Mersereau, 2008] Daniel Adelman and Adam J Mersereau. Relaxations of weakly coupled stochastic dynamic programs. *Operations Research*, 56(3):712–727, 2008.

[Akbarzadeh and Mahajan, 2019] N Akbarzadeh and A Mahajan. Restless bandits with controlled restarts: Indexability and computation of whittle index., 2019.

[Alon *et al.*, 1996] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 20–29, 1996.

[Avrachenkov and Borkar, 2022] Konstantin E Avrachenkov and Vivek S Borkar. Whittle index based q-learning for restless bandits with average reward. *Automatica*, 139:110186, 2022.

[Biswas *et al.*, 2021] Arpita Biswas, Gaurav Aggarwal, Pradeep Varakantham, and Milind Tambe. Learn to intervene: An adaptive learning policy for restless bandits in application to preventive healthcare. *arXiv preprint arXiv:2105.07965*, 2021.

[Bommasani *et al.*, 2021] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[Chaudhuri and Kalyanakrishnan, 2020] Arghya Roy Chaudhuri and Shivaram Kalyanakrishnan. Regret minimisation in multi-armed bandits using bounded arm memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10085–10092, 2020.

[Devlin *et al.*, 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[Dusonchet and Hongler, 2003] Fabrice Dusonchet and M-O Hongler. Continuous-time restless bandit and dynamic scheduling for make-to-stock production. *IEEE Transactions on Robotics and Automation*, 19(6):977–990, 2003.

[Fu *et al.*, 2019] Jing Fu, Yoni Nazarathy, Sarat Moka, and Peter G Taylor. Towards q-learning the whittle index for restless bandits. In *2019 Australian & New Zealand Control Conference (ANZCC)*, pages 249–254. IEEE, 2019.

[Gheshlaghi Azar *et al.*, 2013] Mohammad Gheshlaghi Azar, Rémi Munos, and Hilbert J Kappen. Minimax pac bounds on the sample complexity of reinforcement learning with a generative model. *Machine learning*, 91:325–349, 2013.

[Hawkins, 2003] Jeffrey Thomas Hawkins. *A Langrangian decomposition approach to weakly coupled dynamic optimization problems and its applications*. PhD thesis, Massachusetts Institute of Technology, 2003.

[Hodge and Glazebrook, 2015] David J Hodge and Kevin D Glazebrook. On the asymptotic optimality of greedy index heuristics for multi-action restless bandits. *Advances in Applied Probability*, 47(3):652–667, 2015.

[Jaques *et al.*, 2019] Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, DJ Strouse, Joel Z Leibo, and Nando De Freitas. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International conference on machine learning*, pages 3040–3049. PMLR, 2019.

[Jin *et al.*, 2021] Chi Jin, Qinghua Liu, Yuanhao Wang, and Tiancheng Yu. V-learning–a simple, efficient, decentralized algorithm for multiagent rl. *arXiv preprint arXiv:2110.14555*, 2021.

[Killian *et al.*, 2021a] Jackson A. Killian, Arpita Biswas, Sanket Shah, and Milind Tambe. Q-learning lagrange policies for multi-action restless bandits. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 871–881, New York, NY, USA, 2021. Association for Computing Machinery.

[Killian *et al.*, 2021b] Jackson A. Killian, Andrew Perrault, and Milind Tambe. Beyond "to act or not to act": Fast lagrangian approaches to general multi-action restless bandits. In *AAMAS*, pages 710–718, UK, 2021. AAMAS.

[Killian *et al.*, 2022] Jackson A Killian, Lily Xu, Arpita Biswas, and Milind Tambe. Restless and uncertain: Robust policies for restless bandits via deep multi-agent reinforcement learning. In *Uncertainty in Artificial Intelligence*, pages 990–1000. PMLR, 2022.

[Lefévre, 1981] Claude Lefévre. Optimal control of a birth and death epidemic process. *Operations Research*, 29(5):971–982, 1981.

[Li *et al.*, 2006] Lihong Li, Thomas J Walsh, and Michael L Littman. Towards a unified theory of state abstraction for mdps. In *AI&M*, 2006.

[Liau *et al.*, 2018] David Liau, Zhao Song, Eric Price, and Ger Yang. Stochastic multi-armed bandits in constant space. In *International Conference on Artificial Intelligence and Statistics*, pages 386–394. PMLR, 2018.

[Littman, 1994] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.

[Mate *et al.*, 2021] Aditya Mate, Arpita Biswas, Christoph Siebenbrunner, Susobhan Ghosh, and Milind Tambe. Efficient algorithms for finite horizon and streaming restless multi-armed bandit problems. *arXiv preprint arXiv:2103.04730*, 2021.

[Mate *et al.*, 2022] Aditya Mate, Lovish Madaan, Aparna Taneja, Neha Madhiwalla, Shresth Verma, Gargi Singh, Aparna Hegde, Pradeep Varakantham, and Milind Tambe. Field study in deploying restless multi-armed bandits: Assisting non-profits in improving maternal and child health. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36:12017–12025, 06 2022.

[Nakhleh *et al.*, 2021] Khaled Nakhleh, Santosh Ganji, Ping-Chun Hsieh, I Hou, Srinivas Shakkottai, et al. Neurwin: Neural whittle index network for restless bandits via deep rl. *Advances in Neural Information Processing Systems*, 34:828–839, 2021.

[Papadimitriou and Tsitsiklis, 1999] Christos H Papadimitriou and John N Tsitsiklis. The complexity of optimal queuing network control. *Mathematics of Operations Research*, 24(2):293–305, 1999.

[Ramesh *et al.*, 2021] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[Riquelme *et al.*, 2018] Carlos Riquelme, George Tucker, and Jasper Snoek. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. In *International Conference on Learning Representations*, 2018.

[Shalev-Shwartz and Ben-David, 2014] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[Shapley, 1953] Lloyd S Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10):1095–1100, 1953.

[Sinha and Mahajan, 2022] Amit Sinha and Aditya Mahajan. Robustness of whittle index policy to model approximation. *Available at SSRN 4064507*, 2022.

[Team *et al.*, 2023] Adaptive Agent Team, Jakob Bauer, Kate Baumli, Satinder Baveja, Feryal Behbahani, Avishkar Bhoopchand, Nathalie Bradley-Schmieg, Michael Chang, Natalie Clay, Adrian Collister, et al. Human-timescale adaptation in an open-ended task space. *arXiv preprint arXiv:2301.07608*, 2023.

[Verma *et al.*, 2023] Shresth Verma, Aditya Mate, Kai Wang, Neha Madhiwalla, Aparna Hegde, Aparna Taneja, and Milind Tambe. Restless multi-armed bandits for maternal and child health: Results from decision-focused learning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, pages 1312–1320, 2023.

[Vershynin, 2018] Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

[Wang *et al.*, 2023] Kai Wang, Shresth Verma, Aditya Mate, Sanket Shah, Aparna Taneja, Neha Madhiwalla, Aparna Hegde, and Milind Tambe. Scalable decision-focused learning in restless multi-armed bandits with application to maternal and child health. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12138–12146, 2023.

[Weber and Weiss, 1990] Richard R Weber and Gideon Weiss. On an index policy for restless bandits. *Journal of applied probability*, 27(3):637–648, 1990.

[Whittle, 1988] Peter Whittle. Restless bandits: Activity allocation in a changing world. *Journal of applied probability*, 25(A):287–298, 1988.

[Xie *et al.*, 2020] Qiaomin Xie, Yudong Chen, Zhaoran Wang, and Zhuoran Yang. Learning zero-sum simultaneous-move markov games using function approximation and correlated equilibrium. In *Conference on learning theory*, pages 3674–3682. PMLR, 2020.

[Xiong and Li, 2023] Guojun Xiong and Jian Li. Finite-time analysis of whittle index based q-learning for restless multi-armed bandits with neural network function approximation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[Yaesoubi and Cohen, 2011] Reza Yaesoubi and Ted Cohen. Generalized markov models of infectious disease spread: A novel framework for developing dynamic health policies. *European journal of operational research*, 215(3):679–687, 2011.

[Yu *et al.*, 2020] Wenhao Yu, Jie Tan, Yunfei Bai, Erwin Coumans, and Sehoon Ha. Learning fast adaptation with meta strategy optimization. *IEEE Robotics and Automation Letters*, 5(2):2950–2957, 2020.

[Yu *et al.*, 2022] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35:24611–24624, 2022.

[Zhang and Frazier, 2021] Xiangyu Zhang and Peter I. Frazier. Restless bandits with many arms: Beating the central limit theorem, 2021.

[Zhao *et al.*, 2024] Yunfan Zhao, Nikhil Behari, Edward Hughes, Edwin Zhang, Dheeraj Nagaraj, Karl Tuyls, Aparna Taneja, and Milind Tambe. Towards zero shot learning in restless multi-armed bandits. *AAMAS (2-page extended abstract)*, 2024.

A Additional Experimental Details

A.1 Hyperparameters

In Table 7, we present hyperparameters used, with exceptions (1) for Continuous Synthetic, we use lambda scheduler discount rate = 0.95 (2) for Continuous SIS, we use training opt-in rate = 0.8.

In our experiments, all neural networks have 2 hidden layers each with 16 neurons and tanh activation. The output layer has identity activation and its size is determined by the number of actions (3 for SIS and Continuous SIS, and 2 for other environments).

The $lambda$-network training is similar to that in Killian *et al.*[2022]. After every n_subepochs, we update the λ-network and encourage the actor network to explore new parts of the state space immediately after the $lambda$-update (this exploration is controlled by the temperature parameter that weights the entropy term in the actor loss functions).

Different from Killian *et al.*[2022], we use a λ-network learning rate scheduler, which we found improves the performance and stability of the model.

Table 7: Hyperparameter values.

hyperparameter	value
training opt-in rate	0.8
agent clip ratio	2.0e+00
lambda freeze epochs	2.0e+01
start entropy coeff	5.0e-01
end entropy coeff	0.0e+00
actor learning rate	2.0e-03
critic learning rate	2.0e-03
lambda initial learning rate	2.0e-03
lambda scheduler discount rate	0.99
trains per epoch	2.0e+01
n_subepochs	4.0e+00

A.2 SIS Modeling (Discrete) Experimental Details

Recall that each arm p represents a subpopulation in distinct geographic regions. The state of each arm s is the number of uninfected people within the arm's total population N_p. Transmission within each arm is guided by parameters: κ, the average number of contacts within the arm's subpopulation in each round, and r_{infect}, the probability of becoming infected after contact with an infected person. The probability that a single uninfected person gets infected is then:

$$q = 1 - e^{-\kappa \cdot \frac{S-s}{S} \cdot r_{infect}},$$

where S is the number of possible states, and $s \in [S]$ is the current state. Note $\frac{S-s}{S}$ is the percentage of people who are currently infected. The number of infected people in the next timestep follows a binomial distribution $B(S, q)$.

Recall that there are three available intervention actions a_0, a_1, a_2 that affect the transmission parameters: a_0 represents no action; a_1 represents messaging about physical distancing, dividing κ by a_1^{eff}; a_2 represents the distribution of face masks, dividing r_{infect} by a_2^{eff}. The actions costs are $c = \{0, 1, 2\}$. Following the implementation in [Killian *et al.*, 2022], these parameters are sampled within ranges:

$$\kappa \in [1, 10],\ r_{infect} \in [0.5, 0.99],\ a_1^{eff} \in [1, 10], a_2^{eff} \in [1, 10]$$

A.3 Continuous States Experimental Details

We consider a synthetic dataset with continuous states and binary actions. For the current state s_i of arm i, and action a, the next state s_i' is represented by the transition dynamic:

$$s_i' = \begin{cases} \text{clip}\,(s_i + \mathcal{N}(\mu_{i0}, \sigma_{i0}), 0, 1) & \text{if } a = 0 \\ \text{clip}\,(s_i + \mathcal{N}(\mu_{i1}, \sigma_{i1}), 0, 1) & \text{if } a = 1 \end{cases}$$

Where the transition dynamics are sampled uniformly from the intervals ($\sigma_{i0} = \sigma_{i1} = 0.2$ is fixed):

$$\mu_{i0} \in [-0.5, -0.1], \quad \mu_{i1} \in [0.1, 0.5].$$

We also consider continuous state experiments in real-world settings. In the discrete state SIS Epidemic Model described above, each arm represents a subpopulation, and the state of that arm represents the number of uninfected people within the subpopulation. In real-world public health settings such as COVID-19 control, interventions like quarantine and mask mandates may be imposed on subpopulations of very large sizes such as an entire city. The SIS model is expected to scale to a continuum limit as the population size increases to infinity. Thus, a SIS model with population 1 million would behave roughly similar to that with population 1 billion in terms of the proportions. This notion is inherently captured by continuous models but not by those dealing with absolute numbers.

Following Killian *et al.*[2022], within an arm, any uninfected person will get infected with the same probability. Thus, the number of uninfected people in the next timestep follows a binomial distribution. It is well-known that a normal distribution $\mathcal{N}(\mu, \sigma^2)$ well approximates a binomial distribution $B(n, p)$, with the choice $\mu = np$ and $\sigma^2 = np(1-p)$, when n is sufficiently large.

A.4 Distributional Shift Details

In real-world resources allocation problems, we may observe distribution shifts in arms, i.e., arms in testing are sampled from a distribution slightly different from that in training. In public health settings, a non-profit organization solving RMAB problems to allocate resources to beneficiaries may observe that beneficiaries' behavior or feature information change over time [Wang *et al.*, 2023; Killian *et al.*, 2022]. Additionally, a non-profit organization may have new beneficiaries joining who are in a different subpopulation. In Table 2, we provide ablation results illustrating that **PreFeRMAB is robust to distribution shift in arms**. We measure the shift in distribution using Wasserstein distance. The results demonstrate that even on arm samples from distributions that significantly deviate from that seen in training, PreFeRMAB still achieves strong performance and outperforms baselines.

For each arm, the associated Markov Decision Process (MDP) has only two discrete states, the transition dynamics $p(s'|s, a)$, representing the probability of transitioning to state s' from state s given action a, can be described by four Bernoulli random variables, one for each combination of state and action. By introducing a uniform distribution shift, we can modify the transition probabilities of the Bernoulli random variable associated with each state-action pair by adding

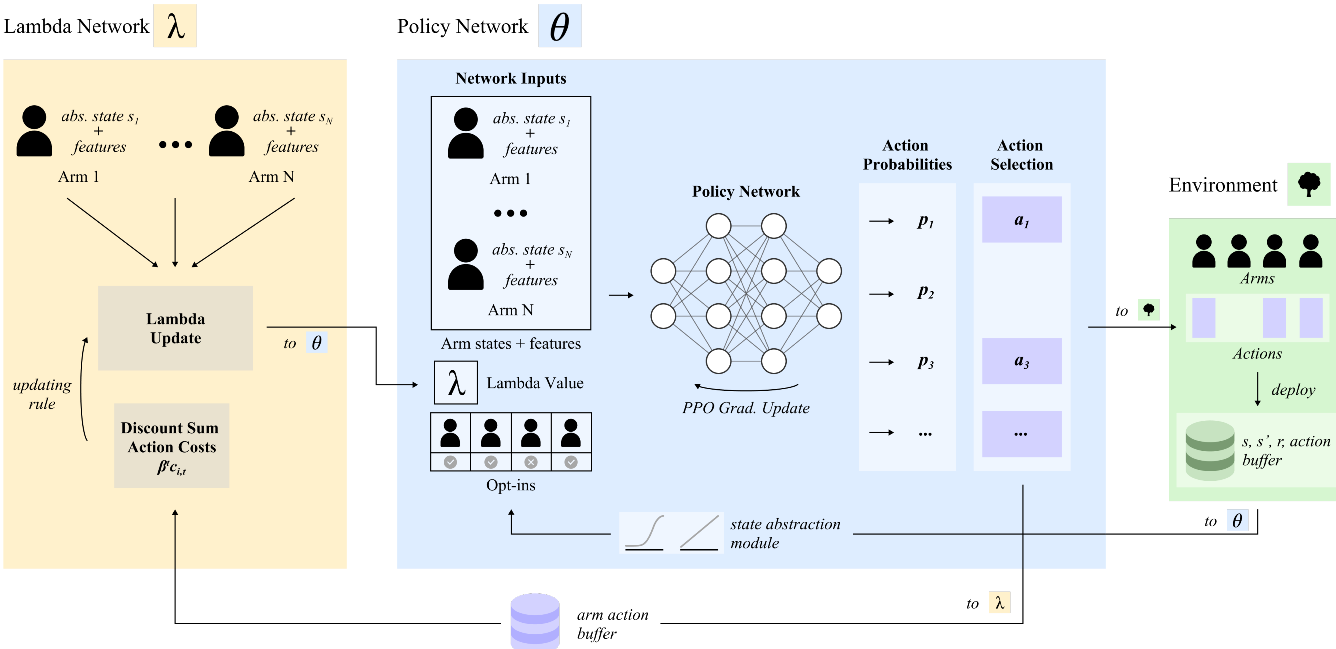

Figure 3: Overview of the PreFeRMAB training procedure. A trained model consists of a policy network, a critic network, a λ-network, and a StateShaping module. Arm states s_i, features z_i, and opt-in decisions ξ are passed through the policy network with an action-charge λ. The policy network independently predicts action probabilities for each arm, which are then greedily selected until the specified budget is reached. These selected actions are used with arm state, feature, and opt-in information to update the λ-network. Updated arm states s' and rewards r from the environment are then added to the buffer, and passed through the state abstraction module before being fed back through the policy network.

a constant δ to the parameter of each Bernoulli distribution. Consequently, this results in a consistent shift in the transition probabilities across all states and actions.

Furthermore, this delta is exactly the Wasserstein distance between the two distributions, which we show here. Suppose we have two discrete probability distributions, P and Q, with their respective probabilities associated with the outcomes x_i and y_j. The The 1-Wasserstein distance, also known as the Earth Mover's Distance $W(P,Q)$, can be calculated by solving the following optimization problem:

$$W(P,Q) = \inf_{\gamma \in \Gamma(P,Q)} \sum_{i,j} |x_i - y_j| \gamma(x_i, y_j),$$

where $\Gamma(P,Q)$ is the set of all joint distributions γ with marginals P and Q, and $\gamma(x_i, y_j)$ represents the amount of "mass" moved from x_i to y_j.

Note $W(P,Q)$ between two Bernoulli distributions with parameters b_1 and b_2 can be succinctly determined as $|b_2 - b_1|$. This is because each Bernoulli distribution has only two potential outcomes, 0 and 1, and so moving mass from one outcome to another across these distributions involves a shift of probability mass $|b_2 - b_1|$ across the one-unit distance between the two points. Therefore, without loss of generality assuming $b_2 \geq b_1$, the Wasserstein distance simplifies to the non-negative difference $b_2 - b_1$.

A.5 Fast Fine-Tuning

In subsection 5.3, we demonstrate that, in addition to strong zero-shot performance, PreFeRMAB may also be used as

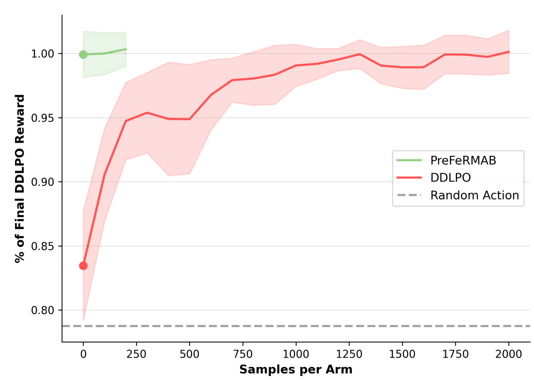

Figure 4: Comparison of the percentage of the final DDLPO (Killian *et al.* [2022] topline) reward achieved by the number of samples per arm. In DDLPO, samples are used for training from scratch; in PreFeRMAB, samples are used to fine-tune a pretrained PreFeRMAB model. Results indicate that PreFeRMAB, from zero-shot results, achieves near-optimal performance, and requires a small fraction of the required DDLPO samples to achieve final DDLPO performance.

a *pretrained model* for fast fine-tuning in specific domains. In particular, we demonstrate that we may start from a pretrained PreFeRMAB model, and train on additional samples for a fixed environment (with fixed arm transition dynamics). We showed that using this pre-trained model can help achieve topline DDLPO performance in significantly fewer fine-tuning samples than required by DDLPO to train from scratch. In Figure 4, we further visualize these results in training curves comparing DDLPO and PreFeRMAB. These training curves, which plot the number of samples per arm against the achieved percentage of final DDLPO reward, are shown for the discrete-state synthetic environment setting for N=21, B=7.0

The results in Figure 4 demonstrate that PreFeRMAB shows both 1) strong zero-shot performance, achieving near-topline reward with *no fine-tuning samples required*, as well as 2) a significant reduction in the number of samples required to achieve final DDLPO performance. In particular, we note that DDLPO, before training, achieves a reward only marginally higher than the average Random Action reward. Alternatively, PreFeRMAB begins, in a zero-shot setting, with a much higher initial reward value. We also observe that PreFeRMAB requires significantly fewer samples per arm to achieve the final DDLPO reward. This is particularly critical in high-stakes, real-world settings where continually sampling arms from the environment may be prohibitively expensive, especially for low-resource NGOs.

A.6 StateShaping

Figure 5 provides a simple example, illustrating how we adapt states through the state abstraction procedure. In this particular example, the reward is an increasing function of the state, and the reward plateaus at state 0.5, i.e. $s \in [0.5, 1]$ achieve the same reward. We map all raw observations in the range $[0.5, 1]$ to abstract state 1. We note that this process is automated, using data collected on arm states and reward from prior (historical) samples to 1) estimate the reward of a current arm, and 2) use this reward to normalize the arm state. We demonstrate how these states are mapped to normalized values in Table 5.

In Figure 5, we show results in two settings after 30 epochs of training, evaluating on a separate set of test arms for zero-shot evaluation. Continuous transition dynamics are used directly as input features for training and evaluation. The results illustrate that state abstraction can help achieve additional performance gains for various challenging reward functions. Drawing from prior literature on state abstraction, this modular component of PreFeRMAB may also serve as a placeholder for future automated state abstraction procedures to improve generalizability and robustness of PreFeRMAB across domains with challenging reward functions.

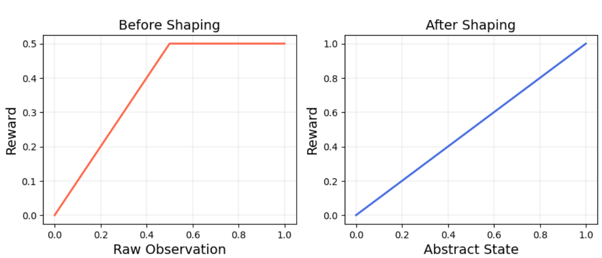

Figure 5: Illustration for StateShaping.

B Ablation Studies

In this section, we provide ablation results over (1) a wider range of opt-in rates than presented in the main paper (Table 12) (2) different feature mappings, including linear and non-linear feature transformations of the original transition probabilities (3) DDLPO topline (Killian *et al.* [2022]) with and without transition probability features as inputs (4) results in more problem settings. The ablation results showcase that PreFeRMAB consistently achieve strong performance and having access to feature information does not provide PreFeRMAB an unfair advantage over DDLPO.

B.1 Opt-in Rates

Number of arms / System capacity	30%	40%	50%	60%	70%
System capacity $N = 21$. Budget $B = 7$.					
No Action	3.09 ± 0.31	3.10 ± 0.32	3.12 ± 0.29	3.14 ± 0.25	3.16 ± 0.22
Random Action	3.57 ± 0.48	3.46 ± 0.48	3.55 ± 0.35	3.55 ± 0.34	3.57 ± 0.33
PreFeRMAB	3.78 ± 0.72	3.75 ± 0.70	4.16 ± 0.57	4.52 ± 0.54	4.45 ± 0.42
System capacity $N = 48$. Budget $B = 16$.					
No Action	3.19 ± 0.27	3.15 ± 0.23	3.13 ± 0.12	3.17 ± 0.12	3.17 ± 0.14
Random Action	3.44 ± 0.26	3.43 ± 0.23	3.46 ± 0.20	3.47 ± 0.17	3.44 ± 0.17
PreFeRMAB	3.90 ± 0.50	3.64 ± 0.30	3.87 ± 0.27	3.85 ± 0.25	4.06 ± 0.31
System capacity $N = 96$. Budget $B = 32$.					
No Action	3.21 ± 0.17	3.17 ± 0.20	3.17 ± 0.15	3.18 ± 0.14	3.17 ± 0.13
Random Action	3.54 ± 0.22	3.55 ± 0.23	3.58 ± 0.18	3.55 ± 0.18	3.56 ± 0.13
PreFeRMAB	3.93 ± 0.33	3.72 ± 0.23	3.79 ± 0.18	4.02 ± 0.21	4.16 ± 0.25

Table 8: Robustness to different opt-in rates with identity mapping. Evaluation follows Table 10: we run for 50 trials with 2 total number of states for each arm, and pretrain a model for each system capacity N and test generalization on different opt-in rates.

Number of arms / System capacity	30%	40%	50%	60%	70%
System capacity $N = 21$. Budget $B = 7$.					
No Action	3.13 ± 0.36	3.19 ± 0.34	3.17 ± 0.27	3.17 ± 0.23	3.15 ± 0.24
Random Action	3.60 ± 0.53	3.62 ± 0.40	3.59 ± 0.34	3.58 ± 0.36	3.57 ± 0.32
PreFeRMAB	3.79 ± 0.49	3.76 ± 0.49	3.79 ± 0.39	3.89 ± 0.39	3.86 ± 0.46
System capacity $N = 48$. Budget $B = 16$.					
No Action	3.15 ± 0.23	3.17 ± 0.21	3.18 ± 0.16	3.18 ± 0.13	3.19 ± 0.14
Random Action	3.41 ± 0.29	3.52 ± 0.27	3.50 ± 0.20	3.49 ± 0.22	3.48 ± 0.17
PreFeRMAB	3.61 ± 0.51	4.08 ± 0.34	4.45 ± 0.30	4.44 ± 0.30	4.44 ± 0.29
System capacity $N = 96$. Budget $B = 32$.					
No Action	3.18 ± 0.18	3.17 ± 0.13	3.18 ± 0.14	3.19 ± 0.14	3.16 ± 0.15
Random Action	3.54 ± 0.22	3.57 ± 0.22	3.55 ± 0.14	3.58 ± 0.15	3.56 ± 0.15
PreFeRMAB	3.74 ± 0.21	3.68 ± 0.23	3.76 ± 0.21	3.98 ± 0.19	4.06 ± 0.22

Table 9: Robustness to different opt-in rates with linear-mappings. Evaluation follows Table 10: we run for 50 trials with 2 total number of states for each arm, and pretrain a model for each system capacity N and test generalization on different opt-in rates.

Throughout the main paper, we provide results for evaluation opt-in rates in the range 80%-100%. In Table 8 and Table 9, we provide ablation results for **opt-in rates in a wider range** of 30%-70%. During the training phase, we maintain an expected opt-in rate of 80%, which may generally range from 70%-90% every training iteration. Given this training configuration, we demonstrate strong results in the main paper for evaluating on a similar range of test-time opt-ins from 80% to 100%. However, we also further demonstrate in Table 8 and Table 9 that our pretrained PreFeRMAB model, despite a training opt-in rate around 80% in expectation, achieves strong results on *testing* opt-in rates from a substantially different range. These results highlight PreFeRMAB's flexibility and ability to generalize to unseen opt-in rates, which may be critical in real-world applications where arms frequently exit and re-enter the environment.

B.2 Feature Mapping

Number of arms / System capacity	80%	85%	90%	95%	100%
System capacity $N = 21$. Budget $B = 7$.					
No Action	3.46 ± 0.20	3.39 ± 0.19	3.40 ± 0.17	3.40 ± 0.18	3.22 ± 0.16
Random Action	3.80 ± 0.31	3.76 ± 0.30	3.79 ± 0.29	3.76 ± 0.31	3.58 ± 0.27
PreFeRMAB	4.57 ± 0.29	4.70 ± 0.33	4.70 ± 0.29	4.64 ± 0.29	4.37 ± 0.25
System capacity $N = 48$. Budget $B = 16$.					
No Action	3.22 ± 0.13	3.28 ± 0.13	3.22 ± 0.12	3.29 ± 0.12	3.21 ± 0.11
Random Action	3.56 ± 0.19	3.65 ± 0.18	3.56 ± 0.18	3.65 ± 0.17	3.57 ± 0.17
PreFeRMAB	3.94 ± 0.23	4.00 ± 0.23	3.86 ± 0.20	3.90 ± 0.16	3.73 ± 0.16
System capacity $N = 96$. Budget $B = 32$.					
No Action	3.24 ± 0.10	3.24 ± 0.10	3.21 ± 0.10	3.21 ± 0.10	3.20 ± 0.10
Random Action	3.61 ± 0.14	3.62 ± 0.15	3.57 ± 0.15	3.57 ± 0.13	3.57 ± 0.14
PreFeRMAB	4.35 ± 0.16	4.36 ± 0.15	4.29 ± 0.14	4.23 ± 0.13	4.20 ± 0.12

Table 10: Results on non-linearly transformed synthetic discrete states. We present final reward divided by the number of arms, averaged over 50 trials with each trial consisting on 10 rounds, for a total of 500 evaluations. The number of states $S = 2$. For each system capacity N, we pretrain a model

In our main paper, we use linear feature mapping, projecting true transition probabilities to features with randomly generated projection matrices. This can be represented by $\mathbf{y} = \mathbf{A}\mathbf{x}$, where $\mathbf{y}$ are the output features, $\mathbf{A}$ is the transformation matrix, and $\mathbf{x}$ denotes the ground truth arm transition probabilities. To demonstrate the robustness of our approach to various types of input features, we also consider **more challenging, non-linear feature mappings**, which may introduce higher representational complexity as compared to linear feature mappings. For these ablation results, we use a sigmoidal transformation, which can be expressed as $\mathbf{y} = \frac{1}{1+\exp(-\mathbf{A}\mathbf{x})}$. We demonstrate the results using these non-linear feature mappings in Table 10. These results indicate that PreFeRMAB consistently outperforms baselines under various forms of feature mappings, and is robust to both linear and non-linear input features.

B.3 DDLPO Topline with Features

Synthetic Experiment	N=21,B=7.0	N=48,B=16.0	N=96,B=32.0
DDLPO, w/o Features	4.63 ± 0.21	4.60 ± 0.18	4.35 ± 0.11
DDLPO, w/ Features	4.63 ± 0.23	4.59 ± 0.17	4.21 ± 0.11

Table 11: Performance comparison of Killian *et al.* [2022] DDLPO topline, with and without ground truth transition probabilities as input features. Results are shown for evaluation on a single, fixed training seed. The results suggest that transition probability features do not significantly improve the final performance of the topline DDLPO model–this implies that PreFeRMAB does not leverage these features for an unfair reward advantage.

In Table 11, we show that having access to **features does not boost the performance of DDLPO**. Features help PreFeRMAB generalize to unseen arms and achieve strong zero-shot results, as demonstrated in the main paper. However, one may ask whether access to these features, as used by PreFeRMAB, may provide an unfair reward advantage over DDLPO, which in its original form [Killian *et al.*, 2022] does not utilize feature information. That is, because input features in our experiments are derived from the original arm transition probabilities, it may be the case that these are used to achieve better performance. To determine whether there is an advantage from utilizing these features, we modify the original DDLPO model to accept ground truth transition probabilities for each arm as feature inputs to the respective policy networks. We present results for DDLPO with and without input features, for a fixed seed, in Table Table 11. In this table, we observe that across synthetic experiments for various system capacities and budgets, DDLPO's performance does not improve given access to features. These results suggest that PreFeRMAB is not leveraging the input features to gain an unfair advantage in evaluation.

B.4 Different values of N, B, S

We present results on a wider range of problem settings, specifically different number of arms N, different budget B, and (for SIS Epidemic Modeling only), different number of possible states S.

Number of arms / System capacity	80%	85%	90%	95%	100%
System capacity $N = 96$. Budget $B = 32$.					
No Action	$3.22_{\pm 0.12}$	$3.24_{\pm 0.12}$	$3.19_{\pm 0.11}$	$3.18_{\pm 0.11}$	$3.18_{\pm 0.11}$
Random Action	$3.62_{\pm 0.13}$	$3.66_{\pm 0.13}$	$3.58_{\pm 0.13}$	$3.60_{\pm 0.12}$	$3.60_{\pm 0.12}$
PreFeRMAB	$4.63_{\pm 0.12}$	$4.71_{\pm 0.12}$	$4.53_{\pm 0.13}$	$4.47_{\pm 0.12}$	$4.61_{\pm 0.10}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$4.58_{\pm 0.13}$
System capacity $N = 48$. Budget $B = 16$.					
No Action	$3.19_{\pm 0.11}$	$3.24_{\pm 0.13}$	$3.18_{\pm 0.11}$	$3.23_{\pm 0.11}$	$3.17_{\pm 0.10}$
Random Action	$3.61_{\pm 0.17}$	$3.70_{\pm 0.21}$	$3.56_{\pm 0.18}$	$3.67_{\pm 0.17}$	$3.58_{\pm 0.16}$
PreFeRMAB	$4.77_{\pm 0.18}$	$4.74_{\pm 0.16}$	$4.62_{\pm 0.19}$	$4.94_{\pm 0.14}$	$4.78_{\pm 0.14}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$4.76_{\pm 0.14}$
System capacity $N = 21$. Budget $B = 7$.					
No Action	$3.44_{\pm 0.21}$	$3.43_{\pm 0.19}$	$3.41_{\pm 0.20}$	$3.38_{\pm 0.17}$	$3.22_{\pm 0.16}$
Random Action	$3.82_{\pm 0.32}$	$3.79_{\pm 0.33}$	$3.77_{\pm 0.31}$	$3.76_{\pm 0.28}$	$3.58_{\pm 0.27}$
PreFeRMAB	$4.20_{\pm 0.27}$	$4.46_{\pm 0.23}$	$4.48_{\pm 0.23}$	$4.74_{\pm 0.26}$	$4.56_{\pm 0.23}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$4.81_{\pm 0.14}$

Table 12: Results on Synthetic with discrete states. We present final reward divided by the number of arms, averaged over 50 trials. For each system capacity N, we pretrain a model. The DDLPO (topline) does not accomodate different opt-in rates and can only be used on 100% opt-in.

Synthetic Evaluation: We first evaluate the performance of PreFeRMAB in the discrete state synthetic environment setting described above. Table 12 illustrates these results. In this synthetic setting, we find that PreFeRMAB is able to consistently outperform Random Action and No Action baselines, and achieve performance comparable to the topline DDLPO approach. Critically, PreFeRMAB achieves good reward outcomes across changing system capacity N, budgets B, as well as different opt-in rates. Additionally, we find that PreFeRMAB achieves *near-topline results from zero-shot learning* in the synthetic setting, compared to the topline DDLPO approach which is trained and evaluated on a fixed set of arm transition dynamics for 100 epochs (we take the best performance of DDLPO across the 100 epochs).

Number of arms / System capacity	**80%**	**85%**	**90%**	**95%**	**100%**
Number of possible states per arm $S = 150$.					
No Action	$5.33_{\pm 0.16}$	$5.30_{\pm 0.15}$	$5.31_{\pm 0.14}$	$5.29_{\pm 0.13}$	$5.28_{\pm 0.13}$
Random Action	$7.03_{\pm 0.17}$	$7.13_{\pm 0.16}$	$7.02_{\pm 0.14}$	$7.11_{\pm 0.13}$	$7.06_{\pm 0.13}$
PreFeRMAB	$8.35_{\pm 0.12}$	$8.38_{\pm 0.11}$	$8.26_{\pm 0.11}$	$8.10_{\pm 0.11}$	$8.00_{\pm 0.10}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$8.09_{\pm 0.11}$
Number of possible states per arm $S = 100$.					
No Action	$5.28_{\pm 0.15}$	$5.20_{\pm 0.13}$	$5.30_{\pm 0.15}$	$5.25_{\pm 0.14}$	$5.27_{\pm 0.15}$
Random Action	$6.95_{\pm 0.19}$	$7.01_{\pm 0.16}$	$7.11_{\pm 0.16}$	$7.06_{\pm 0.15}$	$7.07_{\pm 0.15}$
PreFeRMAB	$7.88_{\pm 0.20}$	$7.91_{\pm 0.19}$	$7.99_{\pm 0.18}$	$8.01_{\pm 0.17}$	$8.02_{\pm 0.16}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$7.99_{\pm 0.08}$
Number of possible states per arm $S = 50$.					
No Action	$5.39_{\pm 0.15}$	$5.47_{\pm 0.15}$	$5.42_{\pm 0.13}$	$5.44_{\pm 0.12}$	$5.46_{\pm 0.12}$
Random Action	$7.29_{\pm 0.17}$	$7.33_{\pm 0.17}$	$7.26_{\pm 0.14}$	$7.38_{\pm 0.15}$	$7.33_{\pm 0.12}$
PreFeRMAB	$8.51_{\pm 0.08}$	$8.37_{\pm 0.11}$	$8.24_{\pm 0.07}$	$8.10_{\pm 0.10}$	$7.93_{\pm 0.09}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$8.04_{\pm 0.08}$

Table 13: Results on SIS Epidemic Model with discrete states. We present final reward divided by the number of arms, averaged over 50 trials. System capacity $N = 20$ and budget $B = 16$. For each number of possible states per arm S, we pretrain a model. The DDLPO (topline) does not accomodate different opt-in rates and can only be used on 100% opt-in.

Number of arms / System capacity	**80%**	**85%**	**90%**	**95%**	**100%**
System capacity $N = 25$. Budget $B = 7$.					
No Action	$2.12_{\pm 0.26}$	$2.30_{\pm 0.29}$	$2.29_{\pm 0.27}$	$2.19_{\pm 0.23}$	$2.26_{\pm 0.25}$
Random Action	$2.86_{\pm 0.32}$	$3.27_{\pm 0.40}$	$3.01_{\pm 0.30}$	$3.09_{\pm 0.35}$	$2.96_{\pm 0.31}$
PreFeRMAB	$5.06_{\pm 0.34}$	$5.26_{\pm 0.33}$	$4.68_{\pm 0.33}$	$4.75_{\pm 0.35}$	$4.61_{\pm 0.27}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$4.68_{\pm 0.09}$
System capacity $N = 25$. Budget $B = 5$.					
No Action	$2.14_{\pm 0.23}$	$2.29_{\pm 0.26}$	$2.24_{\pm 0.28}$	$2.36_{\pm 0.24}$	$2.19_{\pm 0.23}$
Random Action	$2.68_{\pm 0.31}$	$2.95_{\pm 0.36}$	$2.75_{\pm 0.32}$	$2.92_{\pm 0.26}$	$2.69_{\pm 0.21}$
PreFeRMAB	$4.10_{\pm 0.32}$	$4.45_{\pm 0.40}$	$4.39_{\pm 0.33}$	$4.48_{\pm 0.34}$	$3.95_{\pm 0.34}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$4.29_{\pm 0.25}$
System capacity $N = 50$. Budget $B = 10$.					
No Action	$2.27_{\pm 0.24}$	$2.31_{\pm 0.17}$	$2.19_{\pm 0.22}$	$2.21_{\pm 0.21}$	$2.27_{\pm 0.23}$
Random Action	$2.82_{\pm 0.26}$	$2.91_{\pm 0.22}$	$2.72_{\pm 0.23}$	$2.69_{\pm 0.18}$	$2.77_{\pm 0.23}$
PreFeRMAB	$4.21_{\pm 0.30}$	$3.98_{\pm 0.28}$	$3.85_{\pm 0.28}$	$3.68_{\pm 0.28}$	$3.62_{\pm 0.26}$
DDLPO (topline)	*n/a*	*n/a*	*n/a*	*n/a*	$4.08_{\pm 0.26}$

Table 14: Results on ARMMAN with discrete states. We present final reward divided by the number of arms, averaged over 50 trials. For each pair of (N, B), we pretrain a model. The DDLPO (topline) does not accomodate different opt-in rates and can only be used on 100% opt-in.

SIS Evaluation: Next, we evaluate the performance of PreFeRMAB in the discrete-state SIS modelling setting. Table 13 illustrates these results. We evaluate PreFeRMAB for $N = 20, B = 16$ on three different number of possible states per arm $S = 50, 100, 150$, representing the maximum population of a region in the SIS setting. The results shown demonstrate that PreFeRMAB performs well in *zero-shot learning* in settings that model real-world planning problems, especially with larger state spaces and with multiple actions. We again find that PreFeRMAB achieves results comparable to the DDLPO topline with zero-shot testing, compared to DDLPO trained and evaluated on the same constant set of arms.

ARMMAN Evaluation: We next evaluate the performance of PreFeRMAB in the discrete state ARMMAN modeling setting. Table 14 illustrates these results. In these experiments, we show performance for $S = 3$ across 3 training configurations ($(N = 25, B = 5), (N = 25, B = 7), (N = 50, B = 10)$) for 5 test-time opt-in rates. We observe that our approach again performs consistently well in a more *challenging* setting that models real-world planning problems across different system capacities, budgets, and opt-in rates. Specifically, we validate that PreFeRMAB can achieve higher average rewards for increased budgets given a fixed system capacity, which is expected as reward potential increases with higher budgets. Additionally, we see that PreFeRMAB again achieves *zero-shot* results comparable to the DDLPO topline reward, reaching $\sim 90\%$ of the topline reward in zero-shot evaluation.

C Multi-arm Generalization

In the main paper (Table 1), we presented results on Synthetic with $N = 21, B = 7$, demonstrating the benefit of multi-arm generalization. The results are obtained when the Wasserstein distance between training and testing distribution is 0.05 (see Sec A.4 for how we compute the Wasserstein distance). We provide additional results to further showcase the benefits of multi-arm generalization. Specifically, in Table 15, we present results for $N = 12, B = 3$.

	System capacity $N = 12$. Budget $B = 3$.				
# Unique arms	48	39	30	21	12
No Action	$3.11_{\pm 0.31}$	$3.11_{\pm 0.31}$	$3.11_{\pm 0.31}$	$3.11_{\pm 0.31}$	$3.11_{\pm 0.31}$
Random Action	$3.45_{\pm 0.31}$	$3.45_{\pm 0.31}$	$3.45_{\pm 0.31}$	$3.45_{\pm 0.31}$	$3.45_{\pm 0.31}$
PreFeRMAB	$4.35_{\pm 0.28}$	$4.28_{\pm 0.29}$	$4.31_{\pm 0.27}$	$4.04_{\pm 0.32}$	$3.60_{\pm 0.30}$

Table 15: Multi-arm generalization results on Synthetic (opt-in 100%). With the same total amount of data, PreFeRMAB achieves stronger performance when pretrained on more unique arms.

D Proof of Multi-Arm Generalization

In this section, we will shorten n_{epochs} to n for the sake of clarity. In this section, we let C_{sys} to denote a constant which depends on the parameters of the MDP such as budget per arm B/N, cost c_j, discount factor β, $\lambda_{\max}$, $R_{\max}$, D, d and L. It can denote a different constant in every appearance. We list the assumptions made in the statement of the proposition below for the sake of clarity.

Assumption 1. *Suppose the learning algorithm learns neural network weights $\hat{\theta}$, whose policy is optimal for each $(\hat{\mu}_i, \lambda)$ for $i = 1, 2, \ldots, n$ and $\lambda \in [0, \lambda_{\max}]$. That is, it learns the optimal policy for every sample in the training data.*

Assumption 2. *There exists a choice of weights $\theta^* \in \Theta$ which gives the optimal policy for every set of N features $(\hat{\mu})$ drawn as the empirical distribution of i.i.d. samples from μ^* and for every $\lambda \in [0, \lambda_{\max}]$*

Assumption 3. $\Theta = \mathcal{B}_2(D, \mathbb{R}^d)$*, the ℓ_2 ball of radius D in $\mathbb{R}^d$. We assume that*

$$|V(\mathbf{s}, \theta_1, \lambda, \hat{\mu}) - V(\mathbf{s}, \theta_2, \lambda, \hat{\mu})| \leq L\|\theta_1 - \theta_2\|$$

$$|V(\mathbf{s},\theta,\lambda_1,\hat{\mu}) - V(\mathbf{s},\theta,\lambda_2,\hat{\mu})| \leq L|\lambda_1-\lambda_2|$$

Define the population average value function by $\bar{V}(s,\theta) = \mathbb{E}_{\hat{\mu}} \inf_{\lambda\in[0,\lambda_{\max}]} V(\mathbf{s},\theta,\lambda,\hat{\mu})$ and the sample average value function by $\hat{V}(s,\theta) = \frac{1}{n}\sum_{j=1}^{n} \inf_{\lambda\in[0,\lambda_{\max}]} V(\mathbf{s},\theta,\lambda,\hat{\mu})$

Now, consider:

$$\begin{aligned}\bar{V}(s,\hat{\theta}) - \bar{V}(s,\theta^*) &= \bar{V}(\mathbf{s},\hat{\theta}) - \hat{V}(\mathbf{s},\hat{\theta}) + \hat{V}(\mathbf{s},\hat{\theta}) - \hat{V}(\mathbf{s},\theta^*) \\ &\quad + \hat{V}(\mathbf{s},\theta^*) - \bar{V}(\mathbf{s},\theta^*) \\ &= \bar{V}(\mathbf{s},\hat{\theta}) - \hat{V}(\mathbf{s},\hat{\theta}) + \hat{V}(\mathbf{s},\theta^*) - \bar{V}(\mathbf{s},\theta^*) \\ &\geq -2\sup_{\theta\in\Theta}|\bar{V}(\mathbf{s},\theta) - \hat{V}(\mathbf{s},\theta)| \end{aligned} \tag{4}$$

The first step follows by adding and subtracting the same term. In the second step, we have used the fact that Assumptions 1 and 2 imply that $\hat{V}(\mathbf{s},\hat{\theta}) = \hat{V}(\mathbf{s},\theta^*)$. In the third step, we have replaced the discrepancy between the sample averate and the population average at specific points $\hat{\theta},\theta^*$ with the uniform bound over the parameter set Θ.

We use the Rademacher complexity bounds to bound this term. By [Shalev-Shwartz and Ben-David, 2014, Lemma 26.2], we show the following:

Let S denote the random training sample $(\hat{\mu}_1,\dots,\hat{\mu}_n)$ and P_0 denote the uniform distribution $\mathsf{Unif}(\{-1,1\}^n)$. Then, for some numerical constant C, we have:

$$\mathbb{E}_S \sup_{\theta\in\Theta}|\bar{V}(\mathbf{s},\theta) - \hat{V}(\mathbf{s},\theta)| \leq C\mathbb{E}_S\mathcal{R}(\Theta\circ S)$$

Where, $\mathcal{R}(\Theta\circ S)$ is the Rademacher complexity:

$$\begin{aligned}&\mathcal{R}(\Theta\circ S) := \\ &\frac{1}{n}\mathbb{E}_{\sigma\sim P_0}\sup_{\theta\in\Theta}\sum_{i=1}^{n}\sigma_i[\inf_{\lambda} V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \mathbb{E}_{\hat{\mu}}\inf_{\lambda} V(\mathbf{s},\theta,\lambda,\hat{\mu})]\end{aligned}$$

Thus, to demonstrate the result, it is sufficient to show that:

$$\mathcal{R}(\Theta\circ S) \leq \frac{C_{\mathsf{sys}}\mathsf{polylog}(Nn)}{\sqrt{nN}} \tag{5}$$

We will dedicate the rest of this section to demonstrate Equation (5). First we will state a useful Lemma which follows from [Vershynin, 2018, Lemma 1.2.1]

Lemma 1. *Suppose a positive random variable X satisfies: $\mathbb{P}(X>t) \leq A\exp(-\frac{t^2}{2B})$ for some $B>0$, $A>e$ and for every $t\geq 0$ then for some numerical constant C, we have:*

$$\mathbb{E}[X] \leq C\sqrt{B\log A}$$

Proof. From [Vershynin, 2018, Lemma 1.2.1], we have: $\mathbb{E}X = \int_0^\infty \mathbb{P}(X>t)dt$. Thus, we conclude:

$$\begin{aligned}\mathbb{E}X &\leq \int_0^\infty \min(1, A\exp(-\frac{t^2}{2B}))dt \\ &= \sqrt{2B\log A} + \int_{\sqrt{2B\log A}}^\infty A\exp(-\frac{t^2}{2B})dt \\ &= \sqrt{2B\log A} + \int_0^\infty A\exp(-\tfrac{(t+\sqrt{2B\log A})^2}{2B})dt \\ &\leq \sqrt{2B\log A} + \int_0^\infty \exp(-\tfrac{t^2}{2B})dt \\ &\leq \sqrt{2B\log A} + \sqrt{2\pi B} \end{aligned} \tag{6}$$

In the fourth step we have used the fact that $\exp(-(a+b)^2) \leq \exp(-a^2-b^2)$ whenever $a,b>0$. $\square$

Define

$$v_i(\theta) := [\inf_{\lambda} V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \mathbb{E}_{\hat{\mu}}\inf_{\lambda} V(\mathbf{s},\theta,\lambda,\hat{\mu})]\,.$$

We have the following lemma controlling how large v_i is for any given θ.

Lemma 2. *For any $\delta>0$, with probability at-least $1-\delta$,*

$$\sup_i |v_i(\theta)| \leq \sqrt{\frac{C_{\mathsf{sys}}\log(\frac{Nn}{\delta})}{N}}$$

Where C_{sys} depends on the system parameters.

Proof. First, we note that:

$$\begin{aligned}&|\inf_{\lambda} V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \mathbb{E}_{\hat{\mu}}\inf_{\lambda} V(\mathbf{s},\theta,\lambda,\hat{\mu})| \\ &\leq \mathbb{E}_{\hat{\mu}}|\inf_{\lambda} V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \inf_{\lambda} V(\mathbf{s},\theta,\lambda,\hat{\mu})| \\ &\leq \mathbb{E}_{\hat{\mu}}\sup_{\lambda\in[0,\lambda_{\max}]}|V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - V(\mathbf{s},\theta,\lambda,\hat{\mu})| \\ &\leq \sup_{\lambda\in[0,\lambda_{\max}]}|V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \mathbb{E}[V(\mathbf{s},\theta,\lambda,\hat{\mu}_i)]| \\ &\quad + \mathbb{E}_{\hat{\mu}}\sup_{\lambda\in[0,\lambda_{\max}]}|V(\mathbf{s},\theta,\lambda,\hat{\mu}) - \mathbb{E}[V(\mathbf{s},\theta,\lambda,\hat{\mu})]| \end{aligned} \tag{7}$$

In the last step, we have used the fact that $\hat{\mu}$ and $\hat{\mu}_i$ are identically distributed and hence $\mathbb{E}[V(\mathbf{s},\theta,\lambda,\hat{\mu})] = \mathbb{E}[V(\mathbf{s},\theta,\lambda,\hat{\mu}_i)]$. Note that by definition, the value function $V(s,\theta,\lambda,\hat{\mu}_i) = \frac{1}{N}\sum_{j=1}^{N} V(s_j,\theta,\lambda,\mathbf{z}_j)$. Thus, it is clear that $|V(s,\theta,\lambda,\hat{\mu}_i)| \leq A\frac{1+\lambda_{\max}}{(1-\beta)} =: V_{\max}$ where A is a constant which depends on the cost parameters c_j, $\frac{B}{N}$ and the maximum reward. Take $\hat{\mu}_i := (\mathbf{z}_1^{(i)},\dots,\mathbf{z}_N^{(i)})$ and $\hat{\mu} := (\mathbf{z}_1,\dots,\mathbf{z}_N)$.

Thus, for a given λ, we have: $V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \mathbb{E}V(\mathbf{s},\theta,\lambda,\hat{\mu}_i)$ has zero mean and

$$\begin{aligned}&V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \mathbb{E}V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) \\ &= \frac{1}{N}\sum_{j=1}^{N}[V(s_j,\theta,\lambda,\mathbf{z}_j^{(i)}) - \mathbb{E}V(s_j,\theta,\lambda,\mathbf{z}_j^{(i)})] \end{aligned} \tag{8}$$

It is an average of N i.i.d. zero mean random variables, bounded almost surely by $2V_{\max}$. Therefore, using the Azuma-Hoeffding inequality ([Vershynin, 2018]), we have:

$$\mathbb{P}\left(|V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \mathbb{E}V(\mathbf{s},\theta,\lambda,\hat{\mu}_i)| > t\right) \leq C\exp(-\tfrac{c_1 N t^2}{V_{\max}^2})$$

Only in this proof, let $|V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \mathbb{E}V(\mathbf{s},\theta,\lambda,\hat{\mu}_i)| =: H(\lambda)$ for the sake of clarity. Let $B \subseteq [0,\lambda_{\max}]$ be any finite subset. Then, by union bound, we have:

$$\mathbb{P}\left(\sup_{\lambda\in B} H(\lambda) > t\right) \leq C|B|\exp(-\tfrac{c_1 N t^2}{V_{\max}^2}) \tag{9}$$

Suppose B is an ϵ-net for the set $[0,\lambda_{\max}]$ for some $\epsilon > 0$. This can be achieved with $|B| = \frac{\lambda_{\max}}{\epsilon}$. Let $f : [0,\lambda] \to B$ map λ to the closest element in B

$$\begin{aligned}\sup_{\lambda\in[0,\lambda_{\max}]} H(\lambda) &= \sup_{\lambda\in[0,\lambda_{\max}]} H(f(\lambda)) + H(\lambda) - H(f(\lambda)) \\ &\leq \sup_{\lambda\in[0,\lambda_{\max}]} H(f(\lambda)) + 2L\epsilon \\ &\leq \sup_{\lambda\in B} H(\lambda) + 2L\epsilon \end{aligned}\tag{10}$$

Taking $\epsilon = \frac{1}{\sqrt{N}}$, we conclude from Equation (9) that with probability at-least $1-\delta$:

$$\sup_{\lambda\in[0,\lambda_{\max}]} H(\lambda) \leq C_{\mathsf{sys}}\sqrt{\frac{\log(\frac{N}{\delta})}{N}}$$

The same concentration bounds hold for $\sup_\lambda |V(\mathbf{s},\theta,\lambda,\hat{\mu}) - \mathbb{E}V(\mathbf{s},\theta,\lambda,\hat{\mu})|$ and integrating the tails (Lemma 1), we bound obtain the bound:

$$\sup_{\lambda\in[0,\lambda_{\max}]} |V(\mathbf{s},\theta,\lambda,\hat{\mu}) - \mathbb{E}V(\mathbf{s},\theta,\lambda,\hat{\mu})| \leq C_{\mathsf{sys}}\sqrt{\frac{\log N}{N}}$$

Applying a union bound over $i = 1,\dots,n$, conclude the result. □

We state the following folklore result regarding concentration of i.i.d. Rademacher random variables.

Lemma 3. *Given constants $a_1,\dots,a_n \in \mathbb{R}$, and $\sigma_1,\dots,\sigma_n$ i.i.d Rademacher random variables, then for any $\delta > 0$, we have with probability at-least $1-\delta$:*

$$\sum_{i=1}^{n} \sigma_i a_i \leq C\sqrt{\sum_i a_i^2}\sqrt{\log(\tfrac{1}{\delta})}$$

Where C is a numerical constant

We are now ready to prove Equation (5) and hence complete the proof of Proposition 1. Given a data set $\hat{\mu}_1,\dots,\hat{\mu}_n$ and $\theta \in \Theta$, we let $v_i(\theta) := [\inf_\lambda V(\mathbf{s},\theta,\lambda,\hat{\mu}_i) - \mathbb{E}_{\hat{\mu}} \inf_\lambda V(\mathbf{s},\theta,\lambda,\hat{\mu})]$. Given a finite set $\hat{\Theta} := \{\theta_1,\dots,\theta_H\} \subseteq \Theta$, from Lemma 2, we have with probability $1-\delta$,

$$\sup_{\theta\in\hat{\Theta}}\sup_i |v_i(\theta)| \leq \sqrt{\frac{C_{\mathsf{sys}}\log(\frac{nN|\hat{\Theta}|}{\delta})}{N}} =: R(\delta)$$

Therefore, with probability at-least $1-\delta$ over the randomness in $\hat{\mu}_1,\dots,\hat{\mu}_n$, we have:

$$\begin{aligned}&\mathbb{P}\left(\sup_{\theta\in\hat{\Theta}}\sum_{i=1}^{n}\sigma_i v_i(\theta) > t\big|\hat{\mu}_1,\dots,\hat{\mu}_n\right) \\ &\leq C_1|\hat{\Theta}|\exp\left(-\frac{C_2 t^2}{nR^2(\delta)}\right)\end{aligned}\tag{11}$$

We pick $\hat{\Theta}$ to be an ϵ net over Θ. By [Vershynin, 2018, Corollary 4.2.13], we can take $|\hat{\Theta}| \leq (\frac{3D}{\epsilon})^d$. Let $f : \Theta \to \hat{\Theta}$ be the map to its nearest element in $\hat{\Theta}$. Now, we have:

$$\begin{aligned}\sup_{\theta\in\Theta}\sum_i v_i(\theta)\sigma_i &= \sup_{\theta\in\Theta}\sum_i v_i(f(\theta))\sigma_i + [v_i(\theta) - v_i(f(\theta))]\sigma_i \\ &\leq 2n\epsilon L + \sup_{\theta\in\hat{\Theta}}\sum_i v_i(\hat{\theta})\sigma_i\end{aligned}$$

Combining this with Equation (11), we conclude that with $1-\delta$ over the randomness in $\hat{\mu}_1,\dots,\hat{\mu}_n$, we have:

$$\begin{aligned}&\mathbb{P}\left(\sup_{\theta\in\Theta}\sum_{i=1}^{n}\sigma_i v_i(\theta) > t + 2n\epsilon L\Big|\hat{\mu}_1,\dots,\hat{\mu}_n\right) \\ &\leq C_1|\hat{\Theta}|\exp\left(-\frac{C_2 t^2}{nR^2(\delta)}\right)\end{aligned}\tag{12}$$

Taking $\epsilon = \frac{1}{n^{\frac{3}{2}}\sqrt{N}}$ and integrating the tails (Lemma 1), we conclude that with probability at-least $1-\delta$ (with respect to the randomness in $\hat{\mu}_1,\dots,\hat{\mu}_N$).

$$\mathbb{E}[\sup_{\theta\in\Theta}\sum_{i=1}^{n}\sigma_i v_i(\theta)|\hat{\mu}_1,\dots,\hat{\mu}_n] \leq C_{\mathsf{sys}}\frac{R(\delta)}{\sqrt{n}}\mathsf{polylog}(Nn)$$

Define the random variable

$$X := \mathbb{E}[\sup_{\theta\in\Theta}\sum_{i=1}^{n}\sigma_i v_i(\theta)|\hat{\mu}_1,\dots,\hat{\mu}_n]$$

Using the definition of $R(\delta)$, we have:

$$\mathbb{P}(X > t) \leq C_1\exp(-\frac{t^2 nN}{C_{\mathsf{sys}}\mathsf{polylog}(Nn)})\,.$$

We then apply Lemma 1 to the equation above to bound $\mathbb{E}X$ and conclude Equation (5).

E Proof for λ-network Update Rule and Convergence

Proof of Proposition 2. We first consider a simple setting, where the opt-in and opt-out decisions of arms are fixed before training. Taking the derivative of the objective (Eq 2) with respect to λ, we obtain:

$$\frac{B}{1-\beta} - \sum_{i=1}^{N}\mathbb{E}\left[\sum_{\substack{t\in[H]\\ \text{arm } i \text{ opts-in at } t}}\beta^t c_{i,t} + \sum_{\substack{t\in[H]\\ \text{arm } i \text{ opts-out at } t}}\beta^t c_{0,t}\right].$$

Now consider the general case that the opt-in and opt-out decisions are updated at each round during the training. We have

$$\Lambda_t = \Lambda_{t-1} - \alpha \left(\frac{B}{1-\beta} \right) + \alpha \left(\sum_{i=1}^{N} \mathbb{E} \left[\sum_{t=0}^{H} \mathbb{I}\{\xi_{i,t} = 1\} \beta^t c_{i,t} + \mathbb{I}\{\xi_{i,t} = 0\} \beta^t c_{0,t} \right] \right),$$

where the expectation is over the random variables $\xi_{i,t}$ and the action chosen by the optimal policy. Rearranging and simplifying the right hand side terms, we obtain the λ-updating rule. □

Proof of Proposition 3. The proof largely follows the proof of Proposition 2 in Killian *et al.*[2022].

Since the max of piece-wise linear functions is a convex function, Equation 2 is convex in λ. Thus, it suffices to show (1) the gradient estimated using Proposition 2 is accurate and (2) all inputs (states, features, opt-in decisions) are seen infinitely often in the limit. For (1), we note that training the policy network for a sufficient number of epochs under a fixed output of the λ-network ensures that Q-value estimates are accurate. With accurate Q-functions and corresponding optimal policies, the sampled cumulative sum of action costs is an unbiased estimator of expected cumulative sum of action costs. Critically, for the estimator to be unbiased, we do not strictly enforce the budget constraint during training, as in Killian *et al.*[2022]. In inference, we do strictly enforce the budget constraint. For (2), we note that during training, initial states are uniformly sampled, and opt-in decisions are also sampled from a fixed bernoulli distribution.For arms that newly opt-in, the features are uniformly sampled. Thus, both (1) and (2) are achieved. □