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

Yunfan Zhao*1, Nikhil Behari*1, Edward Hughes2, Edwin Zhang1, Dheeraj Nagaraj2, Karl Tuyls2, Aparna Taneja2 and Milind Tambe1,2

1Harvard University
2Google

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 questions 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 $\lambda$-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 adherence, 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 Pretrained Flexible model for RMABs. 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 \( \lambda \)-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]. [Nakhieh 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 \( S_i \) can be either discrete or continuous, and the action space \( A_i \) is a finite set of discrete actions. Each action \( a \in A_i \) has an associated cost \( C_i(a) \), with \( C_i(0) \) denoting a no-cost passive action. The reward at a state is given by a function \( R_i : S_i \to \mathbb{R} \). We let \( \beta \in [0, 1] \) denote a discount factor. Each arm has a unique feature vector \( z_i \in \mathbb{R}^m \) that provides useful information about the arm. Notice our model directly utilizes \( C_i \) in decisions such a policy is how to utilize features following Bellman equation. The key difficulty in learning of notation, we let our algorithms can also be used in the general case where \( \lambda \) may denote the mean and variance of the Gaussian.

For simplicity, we assume that \( S_i, A_i, C_i, \Gamma_i \) 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 \( s \in \mathbb{R}^N \) denote the state over all arms, and we let \( A \in \{0, 1\}^{|A|} \) denote one-hot-encoding of the actions taken over all arms. The agent learns a policy \( \pi \) that maps states \( s \) and features \( z \) to actions \( A \), 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 \( z \) and address opt-in decisions \( \xi \). These are important research questions not addressed in previous works [Killian et al., 2022].

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

s.t. \( \sum_{i=1}^N \sum_{j=1}^{|A|} A_{ij} c_j \leq B \) and \( \sum_{j=1}^{|A|} A_{ij} = 1 \forall i \in [N] \), where \( c_j \in C \) 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 \( R_{\text{max}} \).

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 \( \lambda \)-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 \( \lambda \)-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 \( z_i \) and opt-in decisions \( \xi_i \):

\[
J(s, z, \xi, \lambda^*) = \min_{\lambda \geq 0} \left( \frac{\lambda B}{1 - \beta} + \sum_{i=1}^N \max_{a_i \in [A]} \{Q(s_i, a_i, z_i, \xi_i, \lambda)\} \right),
\]

s.t. \( Q(s_i, a_i, z_i, \xi_i, \lambda) \)

\[
= \xi_i R(s_i) - \xi_i \lambda c_{a_i} + \beta \mathbb{E}[Q(s', a_i, z_i, \xi_i, \lambda) \mid \pi(\lambda)],
\]

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 \( \lambda \). Notice that this relaxation decouples the Q-functions of the arms, and therefore \( Q \) can be solved independently for a given \( \lambda \).

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 \( \lambda \) based on updated opt-in decisions \( \xi \) and features \( z_i \) (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 \( \theta \) and the critic network \( \phi \) (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 expect this model to perform poorly when used out-of-the-box on arms with completely different behaviors.

**Algorithm 1** PreFeRMAB (Training)

1. Input: \( n_{\text{epochs}}, n_{\text{steps}}, \lambda \)-update frequency \( K \in \mathbb{N}^+ \), and system capacity \( N \)
2. Initialize actor \( \theta \), critic \( \phi \), \( \lambda \)-network \( \Lambda \), buffer = [], state \( s \in \mathbb{R}^N \), and features \( z_i \in \mathbb{R}^m \)
3. Initialize StateShaping, and set \( \hat{s} \leftarrow \text{StateShaping}(s) \)
4. for epoch = 1, 2, ..., \( n_{\text{epochs}} \) do
5. Receive opt-in/out requests and update \( \xi \) and \( z_j \).
6. Compute \( \lambda = \Lambda(\hat{s}, \{z_i\}_{i=1}^N, \xi) \)
7. for timestep \( t = 1, \ldots, n_{\text{steps}} \) do
8. for Arm \( i = 1, \ldots, N \) do
9. if Arm \( i \) is opt-in (i.e., \( \xi_i = 1 \)) then
10. Sample an action \( a_i \sim \theta(\hat{s}_i, \lambda, z_i) \)
11. \( s'_i, r_i = \text{Simulate}(s_i, a_i) \)
12. \( s'_i = \text{StateShaping}(s'_i) \)
13. Add tuple \( (s_i, s'_i, a_i, r_i, s'_i, z_i) \) to buffer
14. \( s_i \leftarrow s'_i, \hat{s}_i \leftarrow \hat{s'}_i \)
15. Add tuple \( (\lambda, \xi) \) to buffer
16. Update the \((\theta, \phi)\) pair using buffer.
17. if epoch \( \parallel K = 0 \) then
18. Update \( \Lambda \) via Prop 2 using trajectories in buffer
19. 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., \( z_i \)) be denoted by the probability measure \( \mu^* \). Each ‘sample’ for our policy network training consists of \( N \) features corresponding to \( N \) arms \((z_1, \ldots, z_N)\), drawn i.i.d. from the distribution \( \mu^* \). Call the empirical distribution of \((z_i)\) to be \( \mu \). During training, we receive \( n_{\text{epochs}} \) i.i.d. draws of \( N \) arm features each, denoted by \( \mu_1, \ldots, \mu_{n_{\text{epochs}}} \).

Let \( \Theta \) denote the space of neural network weights of the policy network (for clarity, we shorten the \((\theta, \phi)\) in Algorithm 1 to \( \theta \)). The neural network inputs are Lagrangian multiplier \( \lambda \), state of an arm \( s \), its feature \( z \) and the output is \( a \in \mathcal{A} \). Let \( V(s, \theta, \lambda, \mu) \) denote the discounted reward, averaged over \( N \) arms with features \( \mu \) obtained with the neural network with parameter \( \theta \), starting from the state \( s \) (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 \((\mu_i, \lambda)\) for \( 1 \leq i \leq n_{\text{epochs}} \) and \( \lambda \in [0, \lambda_{\text{max}}] \)
2. There exists \( \theta^* \in \Theta \) which is optimal for every instance \((\mu, \lambda)\).
3. \( \Theta = \mathbb{B}_2(\mathbb{D}, \mathbb{R}^d) \), the \( \ell_2 \) ball of radius \( D \) in \( \mathbb{R}^d \).
4. \( |V(s, \theta, \lambda, \mu) - V(s, \theta, \lambda, \mu^*)| \leq L|\theta - \theta^*| \) and \( |V(s, \theta, \lambda, \mu) - V(s, \theta, \lambda, \mu^*)| \leq L|\lambda - 2\lambda^*| \) for all \( \theta_1, \theta_2, \lambda \in \Theta \)

Then, the generalization error over unseen arms \((\mu, \lambda)\) satisfies:

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

where \( \tilde{O} \) hides polylogarithmic factors in \( n_{\text{epochs}} \), \( N \) and constants depending on \( d, D, L, \beta, \frac{\lambda_{\text{max}}}{N_0} \).

The assumption of existence of \( \theta^* \) is reasonable: This means that there exists a neural network which gives the optimal policy for a family of single-arm MDPs indexed by \((z, \lambda)\). Proposition 1 shows that when \( n_{\text{epochs}} \) 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 is generally more complex. Jointly, the arms form an MDP with \(|S|^N\) 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 [Heshaghli 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 \( \mu^* \) 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 \( \lambda \)-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 \( \lambda \)-network updating rule.

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

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

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

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 $s$, costs $C$, budget $B$, features $z_i \in \mathbb{R}^m$, opt-in decisions $\xi$, agent actor $\theta$, λ-network, StateShaping routine with trained estimator $\hat{r}(\cdot)$
2: Compute $\lambda = \Lambda(s, \{z_i\}_{i=1}^N, \xi)$
3: for Arm $i = 1, \ldots, N$ do
4:   if Arm $i$ is opt-in (i.e. $\xi_i = 1$) then
5:     $\hat{s}_i = \text{StateShaping}(s_i)$
6:     Compute $p_i \sim \theta(\hat{s}_i, \lambda, z_i)$
7:   $a = \text{GreedyProba}(p, C, B)$  // 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. Code is available at https://github.com/yzhao3685/PreFeRMAB

In Appendix B, we provide ablation studies over (1) a wider range of opt-in rates (2) different feature mappings (3) DDLPO tolpline 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 $T_{s=0}^{(i)}$ and $T_{s=1}^{(i)}$ for arm $i$ at states 0 and 1 respectively:

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

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

**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 experimental 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: $\kappa$, 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.

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 [Lefèvre, 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 ARMMA data [Mate et al., 2022]. Our results demonstrate that when features are masked, arms could learn from similar arms’ experience.

<table>
<thead>
<tr>
<th>System capacity $N = 21$, Budget $B = 7$</th>
<th># Unique training arms</th>
<th>45</th>
<th>33</th>
<th>21</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Action</td>
<td>2.88 ± 0.17</td>
<td>2.88 ± 0.17</td>
<td>2.88 ± 0.17</td>
<td></td>
</tr>
<tr>
<td>Random Action</td>
<td>3.25 ± 0.22</td>
<td>3.25 ± 0.22</td>
<td>3.25 ± 0.22</td>
<td></td>
</tr>
<tr>
<td>PreFeRMAB (2/4 Feats. Masked)</td>
<td>3.81 ± 0.23</td>
<td>3.79 ± 0.22</td>
<td>3.55 ± 0.21</td>
<td></td>
</tr>
<tr>
<td>PreFeRMAB (1/4 Feats. Masked)</td>
<td>3.92 ± 0.24</td>
<td>3.70 ± 0.21</td>
<td>3.55 ± 0.20</td>
<td></td>
</tr>
<tr>
<td>PreFeRMAB (0/4 Feats. Masked)</td>
<td>4.02 ± 0.26</td>
<td>3.80 ± 0.22</td>
<td>3.78 ± 0.21</td>
<td></td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Wasserstein Distance</th>
<th>0.05</th>
<th>0.10</th>
<th>0.15</th>
<th>0.20</th>
<th>0.25</th>
</tr>
</thead>
<tbody>
<tr>
<td>System capacity $N = 48$, Budget $B = 16$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No Action</td>
<td>3.09 ± 0.10</td>
<td>2.89 ± 0.08</td>
<td>2.65 ± 0.07</td>
<td>2.49 ± 0.09</td>
<td>2.35 ± 0.07</td>
</tr>
<tr>
<td>Random Action</td>
<td>3.49 ± 0.09</td>
<td>3.25 ± 0.09</td>
<td>2.99 ± 0.16</td>
<td>2.80 ± 0.17</td>
<td>2.57 ± 0.17</td>
</tr>
<tr>
<td>PreFeRMAB</td>
<td>4.50 ± 0.09</td>
<td>4.30 ± 0.10</td>
<td>3.81 ± 0.17</td>
<td>3.79 ± 0.18</td>
<td>3.40 ± 0.12</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Number of arms</th>
<th>80%</th>
<th>85%</th>
<th>90%</th>
<th>95%</th>
<th>100%</th>
</tr>
</thead>
<tbody>
<tr>
<td>System capacity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameters $\alpha_{0i}^{\text{eff}}, \alpha_{1i}^{\text{eff}}$ are uniformly sampled from [2, 8].</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No Action</td>
<td>5.29 ± 0.17</td>
<td>5.29 ± 0.16</td>
<td>5.29 ± 0.16</td>
<td>5.26 ± 0.14</td>
<td>5.28 ± 0.13</td>
</tr>
<tr>
<td>Random Action</td>
<td>6.94 ± 0.15</td>
<td>7.00 ± 0.16</td>
<td>7.05 ± 0.15</td>
<td>6.97 ± 0.14</td>
<td>6.99 ± 0.12</td>
</tr>
<tr>
<td>PreFeRMAB</td>
<td>7.64 ± 0.27</td>
<td>7.75 ± 0.25</td>
<td>7.80 ± 0.18</td>
<td>7.80 ± 0.16</td>
<td>7.82 ± 0.11</td>
</tr>
</tbody>
</table>

| Parameters $\alpha_{0i}^{\text{eff}}, \alpha_{1i}^{\text{eff}}$ are uniformly sampled from [3, 9]. |
| No Action       | 5.29 ± 0.16 | 5.30 ± 0.17 | 5.29 ± 0.15 | 5.26 ± 0.14 | 5.28 ± 0.13 |
| Random Action   | 7.21 ± 0.15 | 7.28 ± 0.18 | 7.29 ± 0.15 | 7.22 ± 0.13 | 7.22 ± 0.12 |
| PreFeRMAB       | 7.77 ± 0.20 | 7.87 ± 0.28 | 7.90 ± 0.22 | 7.95 ± 0.16 | 7.95 ± 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, $\alpha_{0i}^{\text{eff}}, \alpha_{1i}^{\text{eff}}$ 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 mimic the scenario of a real-world non-profit organization using RMABs to allocate resources.

**Continuous State Settings:** Our results (Figure 1) show...
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.

**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%, and 60% respectively.

![Figure 1: Results for Continuous Synthetic domain (N=21,B=7.0) with challenging rewards r(s).](image)

**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.

![Figure 2: Comparison of PreFeRMAB zero-shot performance on unseen arms against that of DDLPO trained and tested on the same set of arms.](image)

**Table 6:** Comparison of PreFeRMAB zero-shot performance on unseen arms against that of DDLPO trained and tested on the same set of arms.

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 suggest 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.

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.

**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.

Acknowledgments
The work is partially supported by Harvard HDSI funding.

Contribution Statement
The first two authors are equal contribution.

References
The first two authors are equal contribution.

Acknowledgments
The work is partially supported by Harvard HDSI funding.

Contribution Statement
The first two authors are equal contribution.

References


David Liu, Zhao Song, Eric Price, and Ger Yang. Stochastic multi-armed bandits in constant


