Inverse Reinforcement Learning for Restless Multi-Armed Bandits with Application to Maternal and Child Health

Gauri Jain[1], **Pradeep Varakantham**[2], **Haifeng Xu**[3], **Aparna Taneja**[4], **Milind Tambe** [1,4]

[1]Harvard University, [2]Singapore Management University, [3]University of Chicago, [4]Google Research
gaurijain@g.harvard.edu, pradeepv@smu.edu.sg, haifengxu@uchicago.edu, {aparnataneja, milindtambe}@google.com,

Abstract

We study restless multi-armed bandits (RMABs) in the context of public health, where there is a need to optimize resource allocation decisions. Until now, RMABs typically solve for the optimal planning policy by assuming the reward function in the problem is fully known. However, in this work, we aim to study whether we can learn the most optimal rewards for an RMAB problem given some demonstrated, ideal behavior. To achieve this, we turn to inverse reinforcement learning (IRL) which is a field of study motivated by the desire to understand and learn the underlying reward structure of an agent's observed behavior. Existing IRL approaches predominantly focus on single agent systems, presenting limitations in dealing with the expansive state spaces characteristic of public health scenarios, where tens of thousands of arms are active simultaneously. We propose a new IRL algorithm specifically for RMAB settings that uses techniques from decision focused learning (DFL) to directly optimize the objective function, allowing for efficient and accurate updates to the learned rewards. We compare our algorithm with the max entropy IRL baseline on runtime and accuracy and find that our algorithm performs better on both metrics. We also propose a framework for how to apply this algorithm in the public health domain where expert trajectories come from domain experts.

1 Introduction

Restless multi-armed bandits (RMABs) (Weber and Weiss 1990; Tekin and Liu 2012) are composed of a set of heterogeneous arms and a planner who can pull multiple arms under budget constraint at each time step to collect rewards. Different from the classic stochastic multi-armed bandits (Gittins, Glazebrook, and Weber 2011; Bubeck and Cesa-Bianchi 2012), the state of each arm in an RMAB can change even when the arm is not pulled, where each arm follows a Markovian process to transition between different states with transition probabilities dependent on arms and the pulling decision. Rewards are associated with different arm states, where the planner's goal is to plan a sequential pulling policy to maximize the total reward received from all arms. RMABs are commonly used to model sequential scheduling problems where limited resources must be strategically assigned to different tasks sequentially to maximize performance. One such application is in healthcare (Mate et al. 2022), where healthcare workers are the planners deciding which beneficiaries (arms) to intervene on (pull).

For RMABs, the prevailing assumption is that planners possess complete knowledge of the rewards associated with different arms (Glazebrook, Ruiz-Hernandez, and Kirkbride 2006). However, there can be scenarios where where the desired trajectories are known, but the explicit rewards for achieving those trajectories remain uncertain. For instance, in the context of pregnancy care which has been studied extensively using RMABs (Mate, Perrault, and Tambe 2021; Wang et al. 2023), a health expert may express an ideal patient's behavior throughout their pre- and postnatal journeys, without precisely being able to quantify rewards for each timestep. To address this challenge, we apply inverse reinforcement learning (IRL) techniques to RMABs, enabling the learning of rewards from observed trajectories, thereby aligning the planner's sequential actions with the desired outcomes.

IRL which was first proposed by (Ng and Russel 2000) is a machine learning technique that aims to infer the underlying reward structure of an agent's behavior by observing its actions in an environment. This is particularly useful in scenarios where the explicit reward function is unknown or difficult to specify, allowing us to learn from demonstrations or expert behavior and subsequently generalize that knowledge to make informed decisions in similar settings. IRL has found applications in various fields, including robotics, autonomous systems, healthcare, and personalized recommendation systems, offering a powerful tool for understanding and replicating expert behavior in complex environments (Arora and Doshi 2021; Skalse and Abate 2023).

In this paper, we study RMAB problems with unknown rewards but with given transition dynamics. The goal is to learn a mapping from states to rewards, which can be used to infer the rewards of unseen RMAB problems to plan accordingly. Prior works (Mate et al. 2022) define the reward function for these problem in advance but this can often be misaligned with the actual goal in the real world. We flip the problem around to use expert trajectories to guide us to find the most appropriate reward function.

There are many existing IRL approaches for a variety of use cases (Arora and Doshi 2021), but none designed specifically for the case of RMABs. Moreover, scalability becomes

critical when the problem size gets higher on the order of 100s or 1000s of arms. Existing work models these problems as a single Markov Decision Process (MDP), but in the RMAB or MAB case, the state space grows exponentially with the number of arms, so we need to design a better approach. Existing IRL techniques also use a large number of sample trajectories to train with, but that is not a realistic scenario in the public health setting where data is limited (Chadi and Mousannif 2021). To remedy these shortcomings, we design an IRL algorithm that uses techniques from decision focused learning (DFL) to efficiently and accurately update learned RMAB reward functions.

Previously, DFL (Wilder, Dilkina, and Tambe 2019) has been proposed to directly optimize the solution quality rather than predictive accuracy, by integrating the one-shot optimization problem (Donti, Amos, and Kolter 2017; Perrault et al. 2020) or sequential problems (Wang et al. 2021; Futoma, Hughes, and Doshi-Velez 2020) as a differentiable layer in the training pipeline. Unfortunately while DFL can successfully optimize the evaluation objective, it is computationally extremely expensive. To address this, (Wang et al. 2023) proposes an approach for decision-focused learning in RMAB problems using Whittle index policy, a commonly used approximate solution in RMABs. We build on this previous work by using this differentiable layer to apply gradient-based updates to rewards. This is the basis of our new IRL algorithm.

Our three key contributions are (i) we design an IRL algorithm for learning RMAB rewards from demonstrated trajectories using decision focused learning; (ii) we show that our algorithm performs better than current baselines in IRL - both in performance and computation time (iii) we propose a framework for applying an IRL algorithm to a large-scale public health setting where expert trajectories come from domain experts.

To establish a benchmark for comparison, we select the Max Entropy IRL algorithm as a commonly used baseline in the IRL literature (Ziebart et al. 2008; Arora and Doshi 2021). By comparing our DFL-based approach against this established method, we demonstrate significant advancements in scalability and performance within the RMAB setting. Our algorithm surpasses the baseline by efficiently handling large state spaces and directly optimizing our desired objective function. These results underscore the importance of our algorithm, offering improved decision-making and resource allocation capabilities in practical applications.

Related Work

RMABs + DFL with missing transition dynamics Our work is inspired by previous work applying DFL to RMAB settings with missing transition dynamics (Wang et al. 2023). This work proposes a novel and scalable decision-focused learning approach using Whittle index policy, establishing its differentiability to optimize the RMAB solution quality directly. By differentiating through the Whittle index policy, they improve the scalability of decision-focused learning in RMAB problems. Their algorithm demonstrates the use of decision-focused learning to real-world RMAB problems with hundreds of arms with significant performance improvements to previous solutions. We build upon this work since we also suffer from the same scalability constraints, and extend this to learn rewards instead of transition dynamics.

Multi-agent IRL (MAIRL) There is some work looking at IRL for multiple agents, but they focus on many fewer agents (Natarajan et al. 2010; Bogert and Doshi 2014), or assume the agents are homogeneous (Šošić et al. 2017) reducing the problem down to a single agent. MAIRL also is used to infer underlying reward structure from observed behavior of multiple agents in competitive settings (Bergerson 2021), which also is not suitable for our case because the arms in our problem do not interact with one another.

Maximum Entropy IRL Max Entropy Inverse Reinforcement Learning (Max Entropy IRL) is a popular approach used to infer the underlying reward function in an environment based on observed expert behavior (Ziebart et al. 2008). The goal of Max Entropy IRL is to find a reward function that not only explains the demonstrated behavior but also maximizes the entropy of the policy distribution. Another way to think of maximum entropy is that it finds the reward distribution that makes minimal commitments beyond constraints, and is therefore least wrong (Arora and Doshi 2021). It has been applied in various domains, including robotics, autonomous driving, and game playing.

We compare our algorithm to the max entropy algorithm due to its strong performance as a baseline in existing IRL literature. Despite it being an older algorithm, it is considered to be one of the most common models (Skalse and Abate 2023) in IRL literature, and therefore an important one to test our multi-agent algorithm against. There are also more recent algorithms like GAIL and AIRL that use deep learning architecture, but they are most effective in continuous state action spaces which is not applicable to our domain. (Fu, Luo, and Levine 2017; Ho and Ermon 2016).

IRL applied to healthcare IRL is a promising method to use in healthcare settings because these settings often involve a sequence of decision-making tasks between a doctor and patient. One such example is using IRL to make decisions on ventilator units and sedatives in ICUs (Yu, Liu, and Zhao 2019). The body of work applying IRL to healthcare domains is growing but there are still a lot of limitations around data scarcity and modeling complex healthcare settings (Chadi and Mousannif 2021). We aim to extend this literature to a more complex public health setting with a large number of heterogenous beneficiaries and budget constraints.

2 Model and Preliminaries

2.1 Restless Multi-armed Bandits

An instance of the restless multi-armed bandit (RMAB) problem is composed of a set of N arms where each is modeled as an independent Markov decision process (MDP). The i-th arm in a RMAB problem is defined by a tuple $(\mathcal{S}, \mathcal{A}, R_i, P_i^{sas'})$. $\mathcal{S}$ and $\mathcal{A}$ are the identical state and action

spaces across all arms. We consider finite state space with $|\mathcal{S}| = M$ fully observable states, and action set $\mathcal{A} = \{0, 1\}$ corresponding to not pulling or pulling the arm, respectively. $P_i^{sas'} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to P$ defines the probability distribution of arm i in state s transitioning to all possible next states $s' \in \mathcal{S}$. $R_i(s)$ is the reward function associated to arm i at the current state s. $R_i(s)$ only depends on the current state and not past or future states.

In a RMAB problem, at each time step $h \in [H]$, the learner observes $\boldsymbol{s}_h = [s_{h,i}]_{i\in[N]} \in \mathcal{S}^N$, the states of all arms. The learner then chooses action $\boldsymbol{a}_h = [a_{h,i}]_{i\in[N]} \in \mathcal{A}^N$ denoting the pulling actions of all arms, which has to satisfy a budget constraint $\sum_{i\in[N]} a_{t,i} \leq K$, i.e., the learner can pull at most K arms at each time step. Once the action is chosen, arms receive action $\boldsymbol{a}_t$ and transitions under P with rewards $\boldsymbol{r}_t = [r_{t,i}]_{i\in[N]}$ accordingly. The total reward is defined by the summation of the discounted reward across T time steps and N arms, i.e., $\sum_{t=1}^{T} \gamma^{t-1} \sum_{i\in[N]} r_{t,i}$, where $0 < \gamma \leq 1$ is the discount factor.

A policy is denoted by π, where $\pi(\boldsymbol{a} \mid \boldsymbol{s})$ is the probability of choosing action $\boldsymbol{a}$ given state $\boldsymbol{s}$. Additionally, we define $\pi(a_i = 1 \mid \boldsymbol{s})$ to be the marginal probability of pulling arm i given state $\boldsymbol{s}$, where $\pi(\boldsymbol{s}) = [\pi(a_i = 1 \mid \boldsymbol{s})]_{i\in[N]}$ is a vector of arm pulling probabilities. We use π^{expert} to denote the optimal policy that knows the true rewards, while π^{learner} to denote a near-optimal policy solver.

2.2 Whittle Index Policy

In this paper, instead of grappling with the optimal policy, we consider the Whittle index policy (Whittle 1988) – the dominant solution paradigm used to solve the RMAB problem. Whittle index policy is easier to compute and has been shown to perform well in practice.

Informally, the Whittle index of an arm captures the added value derived from pulling that arm. The key idea is to determine the Whittle indices of all arms and to pull the arms with the highest values of the index.

To evaluate the value of pulling an arm i, we consider the notion of 'passive subsidy', which is a hypothetical compensation m rewarded for not pulling the arm (i.e. for choosing action $a = 0$). Whittle index is defined as the smallest subsidy necessary to make pulling as rewarding as not pulling, assuming indexability (Liu and Zhao 2010):

Definition 2.1 (Whittle index)**.** Given state $u \in \mathcal{S}$, we define the Whittle index associated to state u by:

$$W_i(u) \coloneqq \inf_m \{Q_i^m(u; a = 0) = Q_i^m(u; a = 1)\} \quad (1)$$

where the value functions are defined by the following Bellman equations, augmented with subsidy m for action $a = 0$.

$$V_i^m(s) = \max_a Q_i^m(s; a) \quad (2)$$

$$Q_i^m(s; a) = m\mathbf{1}_{a=0} + R(s) + \gamma \sum\nolimits_{s'} P_i(s, a, s') V_i^m(s') \quad (3)$$

Given the Whittle indices of all arms and all states $W = [W_i(u)]_{i\in[N],u\in\mathcal{S}}$, the Whittle index policy is denoted by $\pi^{\text{whittle}} : \mathcal{S}^N \longrightarrow [0, 1]^N$, which takes the states of all arms as input to compute their Whittle indices and output the probabilities of pulling arms. This policy repeats for every time step to pull arms based on the index values.

2.3 Soft-top-k Whittle Index Policy

A common choice of Whittle index policy is defined by:

Definition 2.2 (Strict Whittle index policy)**.**

$$\pi_W^{\text{strict}}(\boldsymbol{s}) = \mathbf{1}_{\text{top-k}([W_i(s_i)]_{i\in[N]})} \in \{0, 1\}^N \quad (4)$$

which selects arms with the top-k Whittle indices to pull.

However, the strict top-k operation in the strict Whittle index policy is non-differentiable, which prevents us from gradient based updates in our algorithm, so we use the soft-top-k selection which gives us the probability of pulling each arm. (Xie et al. 2020) (Wang et al. 2021).

We apply soft-top-k to define a differentiable soft Whittle index policy:

Definition 2.3 (Soft Whittle index policy)**.**

$$\pi_W^{\text{soft}}(\boldsymbol{s}) = \text{soft-top-k}([W_j(s_i)]_{i\in[N]}) \in [0, 1]^N \quad (5)$$

Using the soft Whittle index policy, the policy becomes differentiable.

2.4 Inverse RL

We also define a set of J realized trajectories $\mathcal{T} = \{\tau^{(j)}\}_{j\in J}$ generated from a given behavior policy π^{expert} we artificially generate. We denote a full trajectory over H timesteps by $\tau = (\boldsymbol{s}_1, \boldsymbol{a}_1, \cdots, \boldsymbol{s}_H, \boldsymbol{a}_H,)$, where $\boldsymbol{s}, \boldsymbol{a}$ are the joint state and action of all N arms at each timestep. We define a function $\text{Eval}(\pi^{\text{learner}}, \mathcal{T})$ in Equation 7 which we use to update our learned reward estimates. We also propose a method for creating these trajectories directly from previous mobile health data and feedback from public health experts.

3 Problem Statement

This paper studies the RMAB problem where we do not know the reward function $R_i(s)$ in advance, but we know all the other parameters of the problem. Each arm i can have a different reward at each state. We are also given expert trajectories $\mathcal{T}$ that describe optimal behavior that we are trying to mimic with the learned rewards. For now we generate these trajectories using an artificially generated reward R^{expert} to create π^{expert}, but eventually would like to use real world data. The goal is to learn a mapping $R^{\text{learner}} : \mathcal{S}^i \to \mathbb{R}$. The predicted rewards are later used to solve the RMAB problem to derive a policy $\pi^{\text{learner}} = \pi(R^{\text{learner}})$. The performance of the learned policy is evaluated by comparison to the true policy π^{expert}.

4 RMAB IRL

For this work, we take inspiration from recent work in decision-focused-learning for RMABS (Wang et al. 2021), but instead of learning transition dynamics, we learn the reward function for the RMAB.

In our case, we perform iterative updates to the reward function informed by the evaluation function in our IRL algorithm. Figure 1 describes this more closely. In order to apply gradient updates to update the estimated reward, we want to know $\frac{d\text{Eval}}{dR}$. And to compute this, we need to use

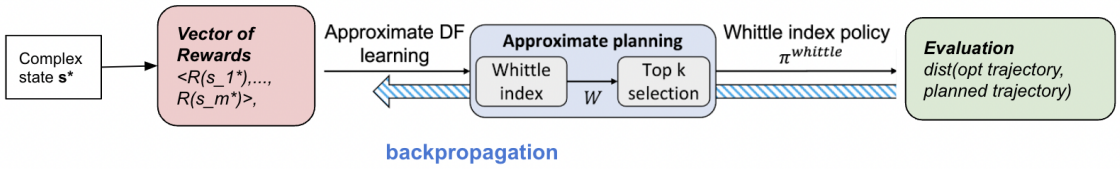

Figure 1: This flowchart visualizes the decision focused learning method of learning rewards. The algorithm iterates through a policy solver using Whittle index policy to estimate the final evaluation and run gradient ascent. From Equation 6, the red box to the blue is $\frac{dW}{dR}$, the blue is $\frac{d\pi^{\text{learner}}}{dW}$, and the blue to green is $\frac{d\text{Eval}(\pi^{\text{learner}},\mathcal{T})}{d\pi^{\text{learner}}}$. Because of Equation 6, we are able to backpropagate through the solver and directly apply $\frac{d\text{Eval}(\pi^{\text{learner}},\mathcal{T})}{dR}$ to update the rewards.

chain rule to make sure all the intermediate steps are differentiable.

$$\frac{d\text{Eval}(\pi^{\text{learner}},\mathcal{T})}{dR} = \frac{d\text{Eval}(\pi^{\text{learner}},\mathcal{T})}{d\pi^{\text{learner}}}\frac{d\pi^{\text{learner}}}{dW}\frac{dW}{dR} \quad (6)$$

where W is the Whittle indices of all states under the learned rewards R. The policy π^{learner} is the Whittle index policy induced by W. This is illustrated in detail in Figure 1.

The term $\frac{d\text{Eval}(\pi^{\text{whittle}},\mathcal{T})}{d\pi^{\text{whittle}}}$ can be computed via policy gradient theorem (Sutton, Barto et al. 1998), and $\frac{d\pi^{\text{whittle}}}{dW}$ was shown to be differentiable in (Wang et al. 2021). However, we still need to show differentiability through Whittle index computation to derive $\frac{dW}{dR}$.

4.1 Differentiability of Whittle Index for Rewards

Whittle indices are often computed using value iteration and binary search (Qian et al. 2016; Mate et al. 2020) or mixed integer linear program. However, these operations are not differentiable and so we need to compute the derivative $\frac{dW}{dR}$ in Equation 6 using a different method.

Lemma 4.1. *(Wang et al. 2023)* $\frac{dW}{dR}$ *is differentiable.*

We use a similar approach from (Wang et al. 2023) to show that $\frac{dW}{dR}$ is differentiable. Using the same linear matrix equation from the proof of $\frac{dW}{dP}$ from that work, we can express whittle indices as a linear function of R allowing for the computation of $\frac{dW}{dR}$ via autodifferentiation.

4.2 Computation Cost and Backpropagation

It is well studied that Whittle index policy can be computed more efficiently than solving the RMAB problem as a large MDP problem. From (Wang et al. 2023) we know that the overall computation of all N arms and M states for a DFL based update is $O(NM^{\omega+1})$ per gradient step. In contrast, the Max Entropy works only on large joint-state MDP's and therefore it's computation cost blows up by $O(M^N)$. Our algorithm significantly reduces the computation cost to a linear dependency on the number of arms N. This significantly improves the scalability of IRL algorithms.

5 Policy Evaluation

In this paper, we use a maximum likelihood estimation (MLE) based evaluation (Arora and Doshi 2021). Given a set of expert trajectories $\mathcal{T}$, at every iteration we estimate $P(\mathcal{T}|\pi^{\text{learner}})$ and use this evaluation to apply a gradient update to the learned rewards (see Equation 7).

We use $s_h^{i,\tau}$ and $a_h^{i,\tau}$ to denote the state and action of arm i in trajectory τ at timestep h. $P(s_{h=1}^{i,\tau})$ is the probability of an armm i being at its initial state at time $h=1$ which is just 1 for us because we start all beneficiaries at state 0. $P(s_h^{i,\tau}, a_h^{i,\tau}, s_{h+1}^{i,\tau})$ is the transition probability $P_i^{sas'}$ defined in Section 2.1. Lastly, $P(a_h^{i,\tau}|\mathbf{s_h^{\emptyset}}, \pi^{\text{learner}})$ is the soft-top-k probability for pulling arm i given π^{learner} which is generated from the learned rewards in that iteration (Equation 5).

Algorithm 1: IRL using Decision-focused Learning in RMAB

Input: $\mathcal{T}, \mathbf{P}$, learning rate r
Initialize: $\mathbf{R} = \mathbf{0}$
for epoch $= 1, 2, \cdots$ **do**
  Compute Whittle indices $W(\mathbb{R})$.
  Let $\pi^{\text{learner}} = \pi_W^{\text{soft}}$ and compute Eval$(\pi^{\text{learner}}, \mathcal{T})$.
  Update $\mathbf{R} = \mathbf{R} + r\frac{d\text{Eval}(\pi^{\text{whittle}},\mathcal{T})}{d\pi^{\text{whittle}}}\frac{d\pi^{\text{whittle}}}{dW}\frac{dW}{dR}$, where $\frac{dW}{dR}$ is computed from Section 4.1.
end for
Return: reward $\mathbf{R}$

$$\begin{aligned}
&\text{Eval}(\pi^{\text{learner}},\mathcal{T}) = P(\mathcal{T}|\pi^{\text{learner}}) \\
&= \prod_{\tau\in\mathcal{T}}\prod_{i\in N} P(s_{h=1}^{i,\tau}) \cdot \prod_{h=1}^{H} P(s_h^{i,\tau}, a_h^{i,\tau}, s_{h+1}^{i,\tau}) \\
&\cdot P(a_h^{i,\tau}|\mathbf{s_h^{\emptyset}}, \pi^{\text{learner}}) \\
&\propto \log P(\mathcal{T}|\pi^{\text{learner}}) \\
&= \sum_{\tau\in\mathcal{T}}\sum_{i\in N} \log(P(s_{h=1}^{i,\tau})) + \sum_{h=1}^{H}\Big[\log(P(s_h^{i,\tau}, a_h^{1,\tau}, s_{h+1}^{i,\tau})) \\
&+ \log(P(a_h^{i,\tau}|\mathbf{s_h^{\emptyset}}, \pi^{\text{learner}}))\Big] \quad (7)
\end{aligned}$$

6 Experiments

Design We perform experiments on a synthetic dataset. We generate $\mathcal{T}$ from randomly generated probabilities $\mathbf{P^{sas'}}$

and rewards $\mathbf{R}$, and while learning we have access to the true $\mathbf{P}^{\mathbf{sas'}}$ values. All experiments are averaged over 10 runs with randomly generated rewards and transition probabilities.

Baseline The baseline we use to determine the success of our algorithm is Max Entropy IRL (Ziebart et al. 2008) which aims to learn the reward that allows for the most variation in behavior while still maximizing the likelihood of seeing the expert trajectories. A key difference between our algorithm and the Max Entropy baseline is that we solve the problem for N independent MDPs while the baseline combines all the arms into a single joint state MDP which grows exponentially with each added arm.

Metrics We compare Max Entropy IRL (**ME-IRL**) with our decision-focused learning based algorithm (**DF-IRL**). To compare performance we compute the l2 norm between the learned soft-k and expert soft-k policy policy probabilites $\|\pi_W^{\text{soft}}(R^{\text{expert}}) - \pi_W^{\text{soft}}(R^{\text{learner}})\|_2$. We can only use this metric on synthetic datasets since we don't have the expert reward in the real world setting. We choose this metric because we want the policy from the learned rewards to mimic the decision making used to create the initial trajectories.

Synthetic datasets We consider two differently sized RMAB problems. The first is composed of $N = 2$ arms, $M = 2$ states, budget $K = 1$, and time horizon $T = 10$ with a discount rate of $\gamma = 0.99$. We use this smaller problem to compare the baseline with our algorithm because the max entropy baseline becomes computationally harder to compute as we increase the number of arms. We also consider an RMAB setting composed of $N = 100$ arms, $M = 2$ states, budget $K = 20$, and time horizon $T = 10$ with a discount rate of $\gamma = 0.99$.

For both settings, the reward function is generated uniformly at random but with the constraint of increasing states having increasing rewards i.e. $\forall i \in$ arms, $\forall m \in$ state space, $R_i(s_m) < R_i(s_{m+1})$. Transition probabilities are also generated uniformly at random but with a constraint that pulling the arm ($a = 1$) is strictly better than not pulling the arm ($a = 0$) to ensure the benefit of pulling.

The historical trajectories $\mathcal{T}$ with $|\mathcal{T}| = J$ are produced by running a random expert policy π^{expert}. The goal is to predict the rewards used to generate the training trajectories.

7 Experimental Results

Improved performance compared to baselines on finding correct rewards In Figure 2, we show the performance of the maximum entropy policy compared to our DFL based algorithm as we increase the J number of trajectories. We can see that our algorithm finds nearly optimal rewards after just 2 sample trajectories, and significantly outperforms the max entropy baseline.

Fast reward learning with few trajectories Figure 3 shows that our DFL based IRL algorithm can learn close to optimal rewards on a much larger problem of $N = 100$ with just 3 input trajectories. Once we move beyond synthetic data, we won't have the $\pi_W^{\text{soft}}(R^{\text{expert}})$ so this experiment validates that our MLE-based Eval function can learn

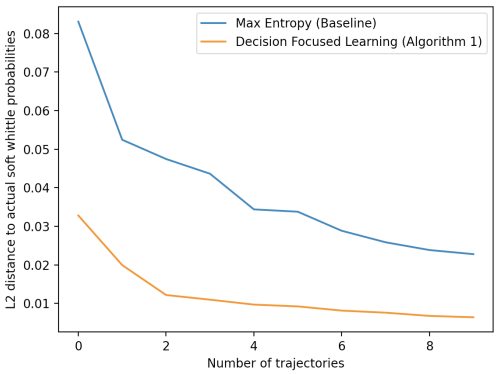

Figure 2: This graph plots the soft-k l2 norm metric (Section 6) for both algorithm as the number of trajectories J is increased. We also set $N = 2, K = 1$. Our algorthm reaches a near optimal reward very quickly and consistently performs better than the max entropy baseline as more trajectories are added

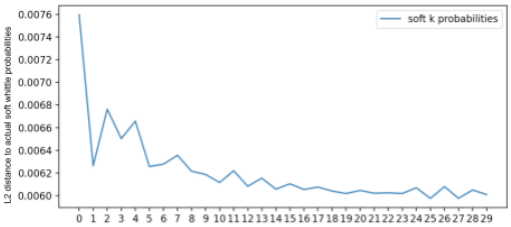

Figure 3: This graph plots the soft-k l2 norm metric (Section 6) for our algorithm for $N = 100, K = 20, J = 3$, epochs $= 30$. Our algorithm learns nearly optimal rewards in approximately 10 epochs even with large numbers of arms.

a nearly optimal reward with very few training trajectories on large problem sizes.

Computation cost comparison Figure 4, compares the computation cost per gradient step of our Whittle index-based decision-focused learning and the max entropy baseline in IRL by changing N (the number of arms) in $M = 2$-state RMAB problem. The max entropy algorithm will not scale to larger problems like maternal and child care with more than 600 people enrolled, while our approach is faster than the baselines with a linear dependency on the number of arms N.

8 Future Applications to Public Health

As the next step of this work, we are interested getting $\mathcal{T}$ from actual healthcare experts. Specifically, we currently work closely with a maternal and child health nonprofit in India named Armman that delivers telehealth care to women during their pregnancy up until the child in one year old. Until now, we have deployed RMAB and DFL based algorithms in the field in collaboration with Armman (Mate et al. 2022; Wang et al. 2023). One way we propose to investi-

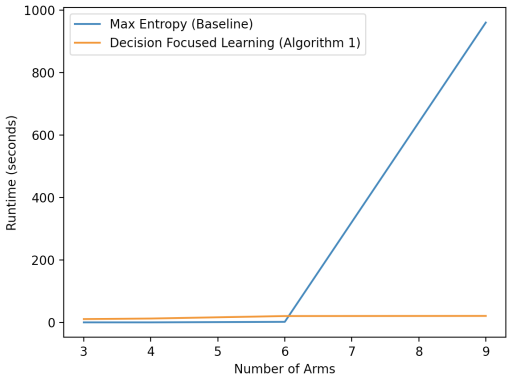

Figure 4: Runtime comparison between max entropy and DFL

gate IRL in this setting is by investigating previous phone engagement data with public health experts to understand what they consider to be good trajectories for different types of beneficiaries (low, medium, high risk). From there we can maximize the likelihood of each type of beneficiary engaging according to the τ assigned to the based on risk score. Through this, we can learn optimal rewards for each beneficiary, which allow us to use those rewards for the future planning problem of deciding when to intervene on them.

Another way we propose to create expert trajectories is by using the actual content delivered in each call. The idea is that there are certain listening patterns that maximize the total new content heard since listening to 60% of the calls amounts to hearing 100% of the health content (Byrne 2020). From these ideal listening patterns, we can again learn rewards that we can use for the actual planning problem to determine how to act on beneficiaries.

9 Conclusion

To the best of our knowledge, this paper presents the first algorithm for using IRL for RMAB problems that is also scalable for large real-world datasets. We also show strong performance in learning rewards with very few input trajectories. Lastly, we propose an approach for applying this work to a real world public health setting.

References

Arora, S.; and Doshi, P. 2021. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297: 103500.

Bergerson, S. 2021. Multi-Agent Inverse Reinforcement Learning: Suboptimal Demonstrations and Alternative Solution Concepts. arXiv:2109.01178.

Bogert, K.; and Doshi, P. 2014. Multi-Robot Inverse Reinforcement Learning under Occlusion with Interactions.

Bubeck, S.; and Cesa-Bianchi, N. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *arXiv preprint arXiv:1204.5721*.

Byrne, F. 2020. Episode 105: Aparna Hegde Founder of ARMMAN.

Chadi, M.-A.; and Mousannif, H. 2021. Inverse Reinforcement Learning for Healthcare Applications: A Survey. In *Proceedings of the 2nd International Conference on Big Data, Modelling and Machine Learning - Volume 1: BML,*, 97–102. INSTICC, SciTePress. ISBN 978-989-758-559-3.

Donti, P. L.; Amos, B.; and Kolter, J. Z. 2017. Task-based end-to-end model learning in stochastic optimization. *arXiv preprint arXiv:1703.04529*.

Fu, J.; Luo, K.; and Levine, S. 2017. Learning Robust Rewards with Adversarial Inverse Reinforcement Learning. *CoRR*, abs/1710.11248.

Futoma, J.; Hughes, M. C.; and Doshi-Velez, F. 2020. Popcorn: Partially observed prediction constrained reinforcement learning. *arXiv preprint arXiv:2001.04032*.

Gittins, J.; Glazebrook, K.; and Weber, R. 2011. *Multi-armed bandit allocation indices*. John Wiley & Sons.

Glazebrook, K. D.; Ruiz-Hernandez, D.; and Kirkbride, C. 2006. Some indexable families of restless bandit problems. *Advances in Applied Probability*, 38(3): 643–672.

Ho, J.; and Ermon, S. 2016. Generative Adversarial Imitation Learning. arXiv:1606.03476.

Liu, K.; and Zhao, Q. 2010. Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access. *IEEE Transactions on Information Theory*, 56(11): 5547–5567.

Mate, A.; Killian, J. A.; Xu, H.; Perrault, A.; and Tambe, M. 2020. Collapsing Bandits and Their Application to Public Health Intervention. In *NeurIPS*.

Mate, A.; Madaan, L.; Taneja, A.; Madhiwalla, N.; Verma, S.; Singh, G.; Hegde, A.; Varakantham, P.; and Tambe, M. 2022. Field Study in Deploying Restless Multi-Armed Bandits: Assisting Non-Profits in Improving Maternal and Child Health. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Mate, A.; Perrault, A.; and Tambe, M. 2021. Risk-Aware Interventions in Public Health: Planning with Restless Multi-Armed Bandits.

Natarajan, S.; Kunapuli, G.; Judah, K.; Tadepalli, P.; Kersting, K.; and Shavlik, J. 2010. Multi-Agent Inverse Reinforcement Learning. In *2010 Ninth International Conference on Machine Learning and Applications*, 395–400.

Ng, A.; and Russel, S. 2000. Algorithms for Inverse Reinforcement Learning. *ICML 2000*.

Perrault, A.; Wilder, B.; Ewing, E.; Mate, A.; Dilkina, B.; and Tambe, M. 2020. End-to-end game-focused learning of adversary behavior in security games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 1378–1386.

Qian, Y.; Zhang, C.; Krishnamachari, B.; and Tambe, M. 2016. Restless poachers: Handling exploration-exploitation tradeoffs in security domains. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, 123–131.

Skalse, J.; and Abate, A. 2023. Misspecification in Inverse Reinforcement Learning. ArXiv:2212.03201 [cs].

Sutton, R. S.; Barto, A. G.; et al. 1998. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Tekin, C.; and Liu, M. 2012. Online learning of rested and restless bandits. *IEEE Transactions on Information Theory*, 58(8): 5588–5611.

Wang, K.; Shah, S.; Chen, H.; Perrault, A.; Doshi-Velez, F.; and Tambe, M. 2021. Learning MDPs from Features: Predict-Then-Optimize for Sequential Decision Making by Reinforcement Learning. *Advances in Neural Information Processing Systems*, 34.

Wang, K.; Verma, S.; Mate, A.; Shah, S.; Taneja, A.; Madhiwalla, N.; Hegde, A.; and Tambe, M. 2023. Scalable Decision-Focused Learning in Restless Multi-Armed Bandits with Application to Maternal and Child Health. ArXiv:2202.00916 [cs].

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

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

Wilder, B.; Dilkina, B.; and Tambe, M. 2019. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1658–1665.

Xie, Y.; Dai, H.; Chen, M.; Dai, B.; Zhao, T.; Zha, H.; Wei, W.; and Pfister, T. 2020. Differentiable top-k operator with optimal transport. *arXiv preprint arXiv:2002.06504*.

Yu, C.; Liu, J.; and Zhao, H. 2019. Inverse reinforcement learning for intelligent mechanical ventilation and sedative dosing in intensive care units. *BMC medical informatics and decision making*, 19(Suppl 2): 57.

Ziebart, B. D.; Maas, A.; Bagnell, J. A.; and Dey, A. K. 2008. Maximum Entropy Inverse Reinforcement Learning.

Šošić, A.; KhudaBukhsh, W. R.; Zoubir, A. M.; and Koeppl, H. 2017. Inverse Reinforcement Learning in Swarm Systems. arXiv:1602.05450.