EXTRACTING EXPERT'S GOALS BY WHAT-IF INTERPRETABLE MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Although reinforcement learning (RL) has tremendous success in many fields, applying RL to real-world settings such as healthcare is challenging when the reward is hard to specify and no exploration is allowed. In this work, we focus on batch inverse RL (IRL) to recover clinicians' rewards from their past demonstrations of treating patients. We explain their treatments based on the what-if future outcomes: "what future would have happened if a different treatment was taken?", and provide interpretability with generalized additive models (GAMs) - a class of accurate, interpretable models. In both simulation and a real-world hospital dataset, our model outperforms baselines and provide explanations consistent with clinical guidelines, while the commonly-used linear model often contradicts them. We also uncover the unreliability of offline metrics such as matched action accuracy to compare IRL methods which is often used in the literature.

1 INTRODUCTION

Reinforcement learning (RL) has achieved tremendous success in many fields including Go (Silver et al., 2017), autonomous driving (Sallab et al., 2017), and healthcare (Chang et al., 2019; Fatemi et al., 2021). However, designing rewards in RL for real-world problems remains challenging when multiple objectives are desired. For example, clinicians often administer vasopressors to increase blood pressure, but a too-high dose might cause vasopressor-induced shock. Also, when robots are designed to navigate to a specific location, the reward function has to prefer not to break nearby items or hurt the people along the path (Amodei et al., 2016). Specifying all possible conditions in the reward is challenging, and designing the magnitude of the reward becomes difficult when multiple goals are needed (e.g. treating patients while reducing the side effects).

One way to avoid reward function tuning is to do imitation learning that directly mimics what experts do by their demonstrations. However, we only extract the rules of how experts act (e.g. administer vasopressors when blood pressure is low) but not the reason why they do (e.g. maintain patient's blood pressure above 65). Therefore, the rules extracted are not suitable for transferring when environments change or different actions are available, while goals recovered from inverse RL (IRL) are more robust and allow the user to inspect and confirm if these are intended consequences.

In many settings such as medicine, experts often behave based on the potential future outcomes: given the current information, what desirable future outcomes would happen if I take certain actions? For example, doctors treat patients with vasopressors to increase their blood pressure in the future. Unlike other IRL methods which use the history to explain the experts' behaviors (e.g. the doctors give vasopressors because the patient's blood pressure is dropping), we instead uses the future outcome of the patients (e.g. the doctors want to maintain the blood pressure above 65 in the next few hours). We believe it's more closer to what doctors think. Importantly, the learned preference is transferable across different environments when actions are different (e.g. different hospitals have different treatment protocols). Finally, it is of interest to clinicians to understand if their behavior match the intended goals, and helps serve as a sanity check tool in reward design.

Generalized Additive Models (GAMs) have been in popular use since the 80s serving as important tools to understand dataset patterns in many fields including healthcare, business and science (Chang et al., 2021). GAMs are also used to audit black-box models (Tan et al., 2018) or discover fairness bias (Tan et al., 2019). As a white-box model, it is surprisingly accurate compared to black-box

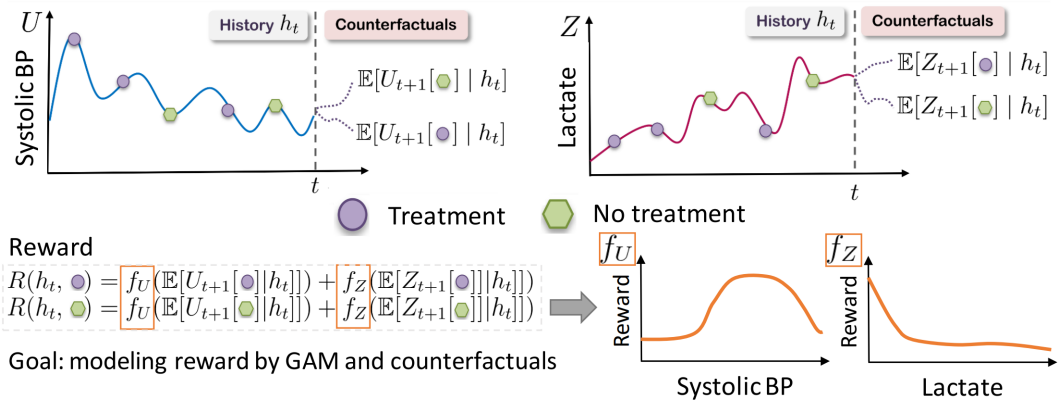

Figure 1: The overview of our work. We first learn a model that predicts future counterfactuals. Then we recover the reward of clinicians by a GAM model based on the estimated counterfactuals. Finally, we interpret what doctors' rewards are from the GAM graphs.

Table 1: The comparison of related works.

Method	Interpretable	What-if reasoning	Reward	Batch data	Have unique solution	Modeled more than 5 features
MMA	✗	✗	$w \cdot \phi(s)$	✗	✗	✓
DSFN	✗	✗	$w \cdot \phi(s,a)$	✓	✗	✓
CIRL	✗	✓	$w \cdot \mathbb{E}[Y_{t+1}\|h]$	✓	✓	✓
AIRL	✗	✗	DNN(s)	✗	✓	✓
iAIRL	✓	✗	DNDT(s)	✓	✓	✗
CAIRL (ours)	✓	✓	GAM($\mathbb{E}[Y_{t+1}\|h]$)	✓	✓	✓

counterparts like deep neural networks (DNNs) for tabular data. To the best of our knowledge, it has not been used in IRL to explain the experts' goals.

In this work, we first predict the potential future outcomes from an observational data by counterfactual modeling in which we make some causal assumptions to identify the effects. Then we use the learned future outcomes to model the clinicians' rewards by an interpretable GAM model in an Adversarial IRL (AIRL) framework, and thus call our model Counterfactual AIRL (CAIRL). In our sepsis simulation, we show CAIRL outperforms both AIRL (Qureshi et al., 2018) and state-of-the-art Counterfactual IRL (CIRL) (Bica et al., 2020) by having a higher accuracy and recovering the underlying rewards better. In a real-world clinical management task (hypotension), our model recovers meaningful clinical thresholds and patterns. However, the linear model such as CIRL, although having comparable offline evaluation metric, often contradicts them and has dissimilar action frequencies from experts. This shows the limitation of the offline evaluation metrics often used in the literature and show the importance of interpretability in IRL for expert inspections.

2 Related Work

Several methods have been proposed to recover the reward function based on expert demonstrations. Max-margin Apprenticeship Learning (MMA, Abbeel & Ng (2004)) assumed the existence of an expert policy π_E that is optimal under some unknown *linear* reward function of the form $R(s,a) = \boldsymbol{w} \cdot \phi(s,a)$ for some reward weights $\boldsymbol{w} \in R^d$ and the feature map $\phi(s,a)$. However, to evaluate how well a policy behaves, they require environmental dynamics to be known. LSTD-Q (Klein et al., 2011) relaxes it by learning to evaluate policy performance via temporal difference method that resembles Q-learning. DFSN (Lee et al., 2019) further improves upon LSTD-Q by using a neural net and prioritized experience replay (Schaul et al., 2015). However, they can only be used to evaluate policies similar to the expert policy. CIRL (Bica et al., 2020) instead learns a counterfactual transition model, and models expert rewards on the estimated future states instead of current states, achieving the state-of-the-art performance. Unfortunately, these MMA methods do not have a unique solution, since even $\boldsymbol{w} = \mathbf{0}$ is a solution to their optimization (Ziebart et al.,

2008). Additionally, the linear assumption in these works is too restrictive for many real-world problems including healthcare, where the goal usually is to maintain patients' vitals in a middle range (e.g. temperature between 36-38) but the linear model only allows monotonically increasing or decreasing relationships.

To solve the non-unique solution problem, Ziebart et al. (2008) proposes Max-Entropy IRL that seeks a reward r to maximize the likelihood of the trajectories under the optimal policy π_E. This formulation has a unique solution unlike MMA, but still assumes the reward is linear. GAIL (Ho & Ermon, 2016) instead formulates Max-Ent IRL as an adversarial game between a policy learner (generator) and a reward model (discriminator) and thus allows non-linear reward modeled by a DNN. However, the reward model may degenerate and not recover the actual expert reward. AIRL (Qureshi et al., 2018) modifies the reward model in GAIL to avoid the degradation and presents a practical scalable implementation in various environments. Although AIRL recovers the reward, the adoption of DNNs in the reward modeling hinders the interpretability. Also, they do not consider the batch clinical setting, which mounts additional challenges of estimating transitions off-policy, restricting policies to stay close to the batch data during learning, and more robust adversarial training procedures due to limited batch data coverage.

iAIRL (Srinivasan & Doshi-Velez, 2020), closest to our work, also aims to recover the clinician's reward and uses an interpretable differential decision tree (DNDT) (Yang et al., 2018) following the AIRL framework. However, due to the exponential feature combinations of DNDT, iAIRL only modeled 5 features. Their performance is also lower than a deep neural network (64% v.s. 71%) while GAM results in similar best accuracy. We summarize prior works in Table 1.

3 BACKGROUND

Markov Decision Process (MDP) We adopt the standard notations of MDP. An MDP consists of a tuple $(S, A, T, T_0, R, \gamma)$ where $s \in S$ current state, $a \in A$ action (discrete, in this work), $s' \in S$ next state, $T(s'|s, a)$ the transition probabilities, and T_0 the initial state distribution, $R(s, a)$ the reward function, and $\gamma \in [0, 1]$ the discount factor. A policy $\pi(a|s)$ gives the probability of taking an action a in a state s. An optimal policy π^* maximizes the cumulative reward G:

$$G_\pi = \sum_{t=0}^{T} E_{s_{t+1}\sim T(s_t,a_t),a_t\sim\pi(s_t)}[\gamma^t r(s_t, a_t)], \quad \pi^* = \text{argmax}_\pi G_\pi$$

In the Batch Inverse Reinforcement Learning (IRL) setting, an agent is given some trajectories (s, a) from a policy which we are told is (near) optimal, and in turn, asked to determine what the reward $R(s, a)$ must have been. Further we assume the "batch" setting which means the agent has no further interaction with the MDP, resembling high-stakes scenarios in real life such as healthcare.

Generalized Additive Models (GAM) GAMs have emerged as a leading model class that is accurate (Caruana et al., 2015), and yet simple enough for humans to understand and mentally simulate how a GAM model works (Hegselmann et al., 2020; Kaur et al., 2020), and is widely used in scientific data exploration (Hastie & Tibshirani, 1995) and model bias discovery (Tan et al., 2018).

GAM are interpretable due to their simple functional forms. Given an input $x \in \mathbb{R}^D$, a label y, a link function g (e.g. g is logits $\log(p/1-p)$ in classification), main effects f_j for each feature j:

$$\textbf{GAM:} \quad g(y) = f_0 + \sum_{j=1}^{D} f_j(x_j).$$

Unlike common models (e.g. DNNs) that use all feature interactions i.e. $y = f(x_1, ...x_D)$, GAM is restricted to not have any feature interaction. This allows 1-D visualization of each f_j independently as a graph i.e. plotting x_j on the x-axis and $f_j(x_j)$ on the y-axis. Note that a linear model is a special case of GAM. GAM is interpretable because human can easily visualize and simulate how it works.

NodeGAM NodeGAM (Chang et al., 2022) is a deep-learning version of GAM that achieves high accuracy and interpretability; its unique differentiability allows its adoption in the AIRL framework.

4 METHODS

Our work builds on Adversarial IRL (Qureshi et al., 2018), in which a discriminator tries to differentiate between the batch expert experiences and the generated experiences. In turn, the generator policy uses the discriminator to create rewards that subsequently improves itself to be much closer to experts. This iterative optimization leads to an equilibrium state where the experiences of the generator and the expert become indistinguishable and the expert reward is recovered. See details of AIRL in Supp. B. We modify two key aspects of AIRL. First, we use the interpretable NodeGAM as the discriminator to generate explainable rewards. Second, the input to the discriminator is the future states s_{t+1} rather than the current state s_t that better resembles the clinicians' reasoning.

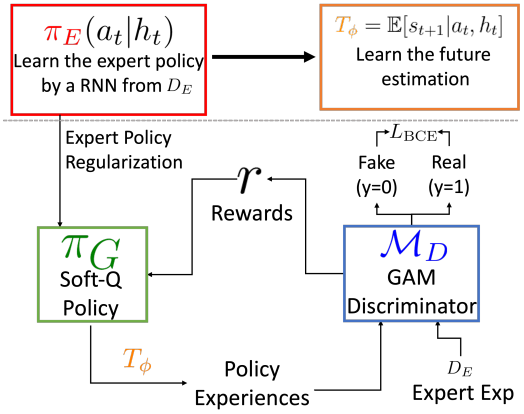

Figure 2: Our system consists of 4 models. First we train an expert policy π_E by a RNN and the future estimator T_ϕ by inverse propensity weighting. Then we train the generator policy π_G and the GAM discriminator $\mathcal{M}_D$ in an adversarial way. Finally we visualize the GAM $\mathcal{M}_D$.

See Fig. 2 for an overview. Below, we introduce how we train our future estimation model (simulator) that estimates the next state s_{t+1}. Then we illustrate how we recover the expert reward in an adversarial framework by training a policy (generator) and a reward model (discriminator) jointly.

Future estimation with batch data Although attributing rewards to the future outcomes better matches how clinicians reason (Futoma et al., 2020), it requires an accurate estimate of what the future looks like. A poor future estimation, for example, a constant, will make all states have the same reward. Additionally, the batch IRL data might be biased. For example, the data could be missing a state-action pair so there is no way to know what that future looks like. Or the dataset could be missing important factors that affect both doctors' treatments and the health outcomes and thus induce wrong associations. Therefore, we make typical causality assumptions to ensure the dataset is "good" by Consistency, Positivity and Unconfoundedness (Rosenbaum & Rubin, 1983; Schulam & Saria, 2017). Please see Supp. A. It often requires domain knowledge to check these assumptions in practice.

Given the assumptions, we can correct for the "treatment bias", which happens because the clincians treat patients non-randomly. For example, a dataset has 20% elderly treated by drug A and 80% for drug B. To avoid this non-randomness, we mimic a dataset that has equal proportion of treatments by stabilized inverse propensity weighting (IPTW) (Robins et al., 2000; Lim et al., 2018). Intuitively, we can remove it by increasing the sample weights of drug A by the inverse of their probability i.e. $\frac{1}{0.2}$ and B by $\frac{1}{0.8}$ to make two treatments have equal samples. Specifically, in each time step t and given the marginal action probability $P(a_t)$, we adjust the sample weight $w_t = P(a_t)/\pi_E(a_t|h_t)$, where π_E is the expert policy (Fig. 1, red) modeled by a RNN. Then we use this sample weight when modeling the future estimation $\mathbb{E}[s_{t+1}|a_t, h_t]$ (Fig. 1, orange). See Supp. F for details.

Generator policy We adopt the state-of-the-art offline RL method as the generator policy: Soft-Q learning (Haarnoja et al., 2017) that allows optimization on both the offline batch data and the online data estimated by the simulator $\mathbb{E}[s_{t+1}|a_t, h_t]$. Specifically, given a network Q, an experience of (s, a, r, s'), and entropy coefficient α, we minimize the Huber loss (smoothed ℓ_1 loss) $L_\mathcal{H}$:

$$\min_Q L_\mathcal{H}((Q(s,a), r(s') + \sum_{a'} \pi(a'|s')(Q(s',a') - \log \pi(a'|s'))) \text{ where } \pi(a|s) = \text{Softmax}(Q(s,a)/\alpha).$$

Experiences (s, a, r, s') come from both offline expert data D_E and online data from $s' \sim \mathbb{E}[s_{t+1}|a_t, h_t]$. To reduce the impact of the inaccurate estimate of $\mathbb{E}[s_{t+1}|a_t, h_t]$, we adopt three ways. First, we only do one-step future predictions from the current state s_t and avoid the multi-step extrapolations like MMA does which accumulates the error in multiple steps. Second, when the action matches the expert action in the batch data, we use the logged next state s_{t+1} instead of the prediction from the model. Finally, we find a smaller weight $\delta = 0.5$ on the loss of simulated data produces the best result. Specifically, given the expert batch data D_E and the estimation model T:

$$L = \mathbb{E}_{D_E}[L_\mathcal{H}(s, a, s')] + \delta \mathbb{E}_{s \sim D_E, a \sim \pi(\cdot|s), s' \sim T(s,a)}[L_\mathcal{H}(s, a, s')] \quad (1)$$

Table 2: The performance of 7 IRL models in the sepsis simulation. The method is better with higher culmulative reward and lower distance (Dist) to the ground truth reward in GAM graphs. The best numbers are bolded. Linear models (MMA, CIRL, Linear-AIRL/CAIRL) can not fit the GAM MDP and perform the worst. GAM-CAIRL performs better than GAM-AIRL.

	$\gamma = 0.9$				$\gamma = 0.5$			
	GAM MDP		Linear MDP		GAM MDP		Linear MDP	
	Reward ↑	Dist ↓	Reward ↑	Dist ↓	Reward ↑	Dist ↓	Reward ↑	Dist ↓
MMA	-6.112 ± 0.027	-	1.631 ± 0.004	-	-1.081 ± 0.010	-	0.316 ± 0.001	-
CIRL	-7.637 ± 0.040	-	1.629 ± 0.013	-	-1.111 ± 0.001	-	0.341 ± 0.003	-
Linear-AIRL	-	-	1.690 ± 0.011	0.051	-	-	0.343 ± 0.002	0.358
FCNN-AIRL	-0.919 ± 0.012	-	1.664 ± 0.010	-	-0.362 ± 0.003	-	**0.344** ± 0.003	-
GAM-AIRL	-0.931 ± 0.012	0.358	1.670 ± 0.013	0.210	**-0.357** ± 0.003	0.421	0.342 ± 0.003	0.234
Linear-CAIRL	-6.449 ± 0.008	0.471	**1.698** ± 0.032	**0.016**	-1.003 ± 0.012	0.547	0.343 ± 0.003	0.343
FCNN-CAIRL	-0.947 ± 0.010	-	1.687 ± 0.009	-	**-0.357** ± 0.004	-	0.343 ± 0.004	-
GAM-CAIRL	**-0.894** ± 0.013	**0.282**	1.682 ± 0.022	0.073	**-0.357** ± 0.001	**0.345**	**0.344** ± 0.004	**0.195**
Expert	-0.883 ± 0.002	0.000	1.708 ± 0.008	0.000	-0.356 ± 0.009	0.000	0.345 ± 0.005	0.000

Behavior Cloning regularization To stabilize the generator optimization, we find that it's crucial to regularize the early part of optimization to be close to the expert policy derived from behavior cloning (BC) (i.e. using supervised learning to predict actions). When updating the Q-network, we add an additional KL divergence loss between the current policy π and expert policy π_{bc} i.e. $L_{bc} = \lambda_{bc} KL(\pi_Q, \pi_{bc})$. We linearly decay λ_{bc} to 0 in the first half of the training.

Discriminator: the reward model We follow the similar design from AIRL that trains a binary classifier to predict whether feature map ϕ comes from the expert or the generator, but here ϕ is the next state s_{t+1} instead of s_t. Specifically, given g as the reward model, h as the shaping term modeling in AIRL, π the generator's policy, the discriminator logit D is:

$$D(s, a, s') = g(\phi) + h(s') - h(s) - \log \pi(s, a)$$

And we set ϕ as s' while AIRL sets ϕ as s. We set both g and h as the NodeGAM, and the class y of expert data as 1 and generated data as 0, and optimize the binary cross entropy loss (BCE):

$$L_D = \mathbb{E}_{s,a,s' \sim D_E}[\text{BCE}(D(s, a, s'), \mathbf{1})] + \mathbb{E}_{s \sim D_E, a \sim \pi(\cdot|s), s' \sim T(s,a)}[\text{BCE}(D(s, a', s'), \mathbf{0})].$$

Discriminator Stabilizing Tricks When optimizing discriminator, we use both one-sided label smoothing (Salimans et al., 2016) and add a small Gaussian noise (Jenni & Favaro, 2019) to the inputs which have been shown useful to stabilize GAN adversarial optimization (see Supp. G.1).

Reward scaling Since the rewards can be arbitrarily shifted and scaled without changing the resulting optimal policy, we scale them into similar value range on the graphs. See Supp. H for details.

5 RESULTS

We evaluate our model on two tasks: a simulated sepsis task and a real-world clinical treatment task.

Baselines We compare with the widely-used Max-Margin Apprenticeship learning (MMA) that follows the linear design of the reward. We also compare with the counterfactual version of MMA, CIRL, in our simulations. In addition, we also compare with the AIRL framework that uses the current state s_t instead of our what-if reasoning of the future outcome s_{t+1}. Within both AIRL and CAIRL frameworks, we compare 3 reward models: (1) Linear, (2) NodeGAM (GAM), and (3) Fully-Connected Neural Network (FCNN).

5.1 SEPSIS: A CLINICAL SIMULATOR TO COMPARE METHODS UNDER SPECIFIED REWARDS

We first experiment on a challenging sepsis simulation environment from Oberst & Sontag (2019). This is a coarse physiological model for sepsis with 4 time-varying vitals (Systolic BP, Percentage

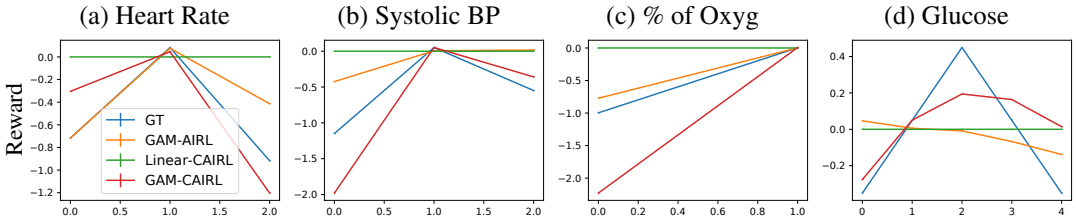

Figure 3: The 4 shape plots in the sepsis dataset where rewards are modeled by a GAM model (GAM MDP). Our GAM-CAIRL (red) is closest to the ground truth (GT, blue), while Linear (green) model can not handle non-linear reward and thus act as a straight line.

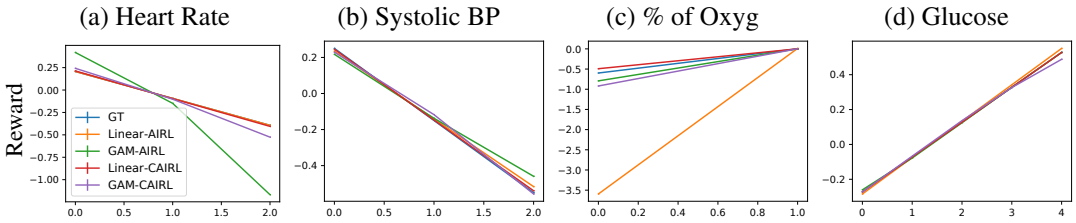

Figure 4: The 4 shape plots in the sepsis dataset where rewards are modeled by a Linear model (Linear MDP). All 4 models are close to GT (Blue) except the GAM-AIRL (green) in (a) HR and the Linear-AIRL (orange) in % of Oxyg.

of Oxygen, ...) that's discretized (e.g. "low"/"normal"/"high"). Combined with 3 different binary treatments (total 8 actions) and 1 static variable (diabetes), our resulting MDP consists of 1440 possible discrete states. Trajectories are at most 20 timesteps. In addition to 4 vitals, in our feature space, we include a uniform noise feature to make it harder. Since this simulator is a discrete environment, we can solve the exact optimal policy via value iteration to generate expert data. We also use the underlying MDP as our future estimation model that resembles a good trained model. We generate 5000 trajectories with optimal policy for both training and test data.

To test if our model can recover the ground truth reward, we design the reward function in two forms. (1) GAM MDP: we model the reward as an additive function of s_{t+1}, i.e. $r = \sum_j f_j(s_{(t+1)j})$. (2) Linear MDP: we model the reward as a linearly additive function of s_{t+1} i.e. $r = \boldsymbol{w} \cdot \boldsymbol{s}_{t+1}$. Its specific functional form can be found in Supp. E. Note that the rewards may not be clinically meaningful, but they allow us to quantitatively compare different methods.

In Table 2, we show the reward and the distance to ground truth reward of all 8 models under $\gamma = 0.9$ and 0.5. First, in GAM MDP where ground truth is non-linear, we see MMA, CIRL and Linear expectedly perform poorly because of their linear nature. Out of all models, GAM-CAIRL achieves the highest reward and also recovers ground truth more faithfully with the lowest distance on the shape graph to the ground truth (Fig. 3). It also outperforms GAM-AIRL, which does not include what-if reasoning but still achieves reasonable performance. In Linear MDP, we see Linear-CAIRL performs the best. Linear-AIRL has a similar reward but its distance on the graph is much larger (0.051 v.s. 0.016). MMA and CIRL perform slightly worse than Linear-AIRL and Linear-CAIRL. GAM and FCNN perform well without significant differences from Linear-CAIRL.

We repeat the experiment with $\gamma = 0.5$ and find the reward difference between models becomes smaller. In GAM MDP, GAM-CAIRL still achieves the smallest distance to the ground truth. In Linear MDP, however, GAM has a smaller distance than Linear in both CAIRL and AIRL settings; we find Linear has an opposite slope in feature Glucose that leads to a larger distance.

We visualize our shape graphs in GAM MDP when $\gamma = 0.9$ in Fig. 3. First, Linear as expected can not capture the non-linear relationship and thus is flat. In (a) Heart Rate, (b) Systolic BP and (c) % of Oxygen, all models except Linear capture the correct trend. For (d) Glucose, only GAM-CAIRL captures the correct shape that finds value 2 produces the highest reward. In Fig. 4, we show the shape graphs in Linear MDP. All models capture the correct trend in all 4 features.

Table 3: The test accuracy (with stdev) of policy actions matched to experts. BC is behavior cloning.

	BC	Linear-AIRL	GAM-AIRL	FCNN-AIRL	Linear-CAIRL	GAM-CAIRL	FCNN-CAIRL
Acc(%)	72.0 $\pm$ 1.0	74.1 $\pm$ 0.4	74.2 $\pm$ 0.9	73.8 $\pm$ 0.5	**74.8** $\pm$ 0.4	**74.7** $\pm$ 0.3	74.4 $\pm$ 0.4

5.2 MIMIC3 Hypotension Treatment Dataset

To demonstrate the utility of our method, we experiment on a real-world medical decision making task of managing hypotensive patients in the ICU. Hypotension is correlated with high mortality (Jones et al., 2004). Although there exists various clinical guidelines (Bunin; Khanna, 2018), there is no standardized treatment strategy since there are many underlying causes of hypotension (Srinivasan & Doshi-Velez, 2020).

Preprocessing We use MIMIC-III (Johnson et al., 2016), filtering to adult patients with at least 2 treatments within the first 72 hours into ICU resulting in 9,404 ICU stays. We discretize trajectories into 2-hour windows, so trajectories end either at ICU discharge or at 72 hours into the ICU admission with at most 36 timesteps and 35 actions taken. We follow the preprocessing of Futoma et al. (2020) to select two treatments: fluid bolus therapy and vasopressors. We discretize both treatments into 4 levels (none, low, medium and high). We extract 5 covariates and 29 time-varying features and impute missing values with the forward imputation. For each model, we perform 5-fold cross validation with each fold having 60-20-20 for train-val-test splits. We set $\gamma = 1$. Please see Supp. C.

Following previous works, we compare the accuracy of the actions matched to the expert in Table 3. Note that this only evaluates how good the policy (generator) matches experts under experts' states distributions but does not reflect how good the learned reward (discriminator) is. We also compare with behavior cloning (BC) which does the supervised learning to predict actions from the logged expert data. We find both Linear-CAIRL and GAM-CAIRL perform the best although the difference to AIRL is not significant. All methods outperform BC.

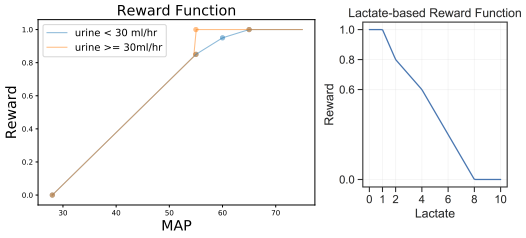

Figure 6: The rewards designed by clinicians (Futoma et al., 2020).

In Fig. 5, we evaluate the shape graphs derived from GAM-CAIRL, Linear-CAIRL, and GAM-AIRL. First, in Fig. 6, we show the clinician-designed reward for treating hypotensive patients (Futoma et al., 2020) as our ground truth of two features: MAP and Lactate. We find GAM-CAIRL recovers the right regions: in Fig. 5(a) the reward increases as MAP increases above 65, which matches the normal range of MAP above 65 in Fig. 6. Surprisingly, our method recovers a correct trend that MAP bigger than 100 indicating hypertension should also have low rewards, but clinicians forget to specify in Fig. 6(a); if uncaught, such reward can result in risky RL policies that always increase the MAP at all cost. In Fig. 5(b), our model also matches the trend in Fig. 6(b) by substantially dropping the reward as lactate value grows beyond 2 and saturating at 10. Although Linear-CAIRL has similar accuracy to GAM-CAIRL (Table 3), it shows an opposite pattern from the clinicians' understanding by having an increasing trend in both MAP and Lactate. Finally, GAM-AIRL also fails to learn the reward should be low when MAP ≤ 65 in (a).

We illustrate other features (c)-(d). In Systolic BP (c), since the goals of both fluids and vasopressors management are to increase blood pressure, it makes sense that the lower blood pressure has a lower reward and GAM successfully captures that. Unfortunately, GAM assigns high reward to high Systolic BP even when it is around 200 which we think is an artifact due to the inductive bias of tree-based Node-GAM that remains flat. Linear model instead indicates a negative slope that suggests the lower the blood pressure the better which is clearly in violation of the goal of treating hypotension. Glasgow Coma Scale (GCS, (d)) describes the level of consciousness of patients with value 15 meaning high consciousness and 3 meaning deep coma. Our GAM model captures this notion by learning a steady increase of reward as GCS increases, while the linear model learns the opposite trend which does not make sense.

In the second row (e)-(h), PO2 (e) measures the oxygen concentration in blood. Studies show that PO2 > 100 leads to hyperoxemia which is associated with higher mortality. Shape plots show a sharp decrease when PO2 is right above 100 which confirms this. It is important to note that low PO2

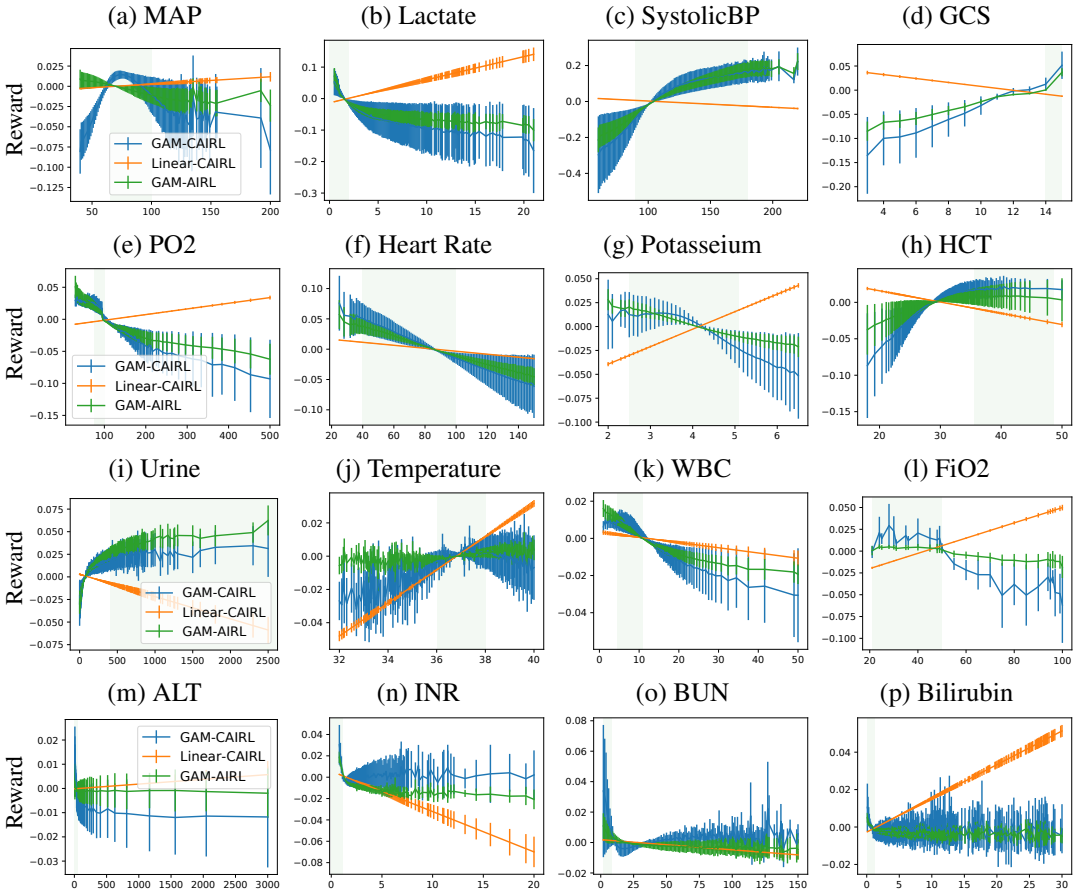

Figure 5: The 16 (out of 29) shape plots of GAM-CAIRL v.s. Linear-CAIRL v.s. GAM-AIRL on MIMIC3 Hypotension management tasks. The pale green region shows the normal range of values based on clinical guidelines (Supp. I) and thus should have higher reward. GAM mostly aligns to the guidelines while the Linear model often violates them. GAM-AIRL are mostly similar to GAM-CAIRL except for (a), (j) and (l) where it matches guidelines worse.

(<80) should not be rewarded either, suggesting that the high reward learned by GAM at PO2<80 is likely another artifact. Again, Linear model learns the completely opposite trend in PO2. For heart rate (f), both GAM and Linear model correctly agree that slower heart rate is generally better although the heart rate of 20 is likely too low and should not be rewarded. For potassium (g), high potassium is correlated with kidney disease and sometimes can cause a heart attack or death. Our GAM roughly matches the clinical guideline that assigns a higher reward for normal range (2.5-5.1) and assigns a much lower reward for high potassium. Instead linear model has the opposite trend again. In (h), hematocrit level (HCT) is the percentage of red cells in the blood, and normally when hematocrit is too low it indicates an insufficient supply of healthy red blood cells (anemia). GAM successfully captures this by assigning high reward in HCT but Linear model again contradicts the clinical knowledge. Fig. (i)-(p) have similar findings and thus we defer the descriptions to Supp. D.

Action Frequencies Visualization In Fig. 7, we visualize the action frequencies of the (a) experts, (b) GAM-CAIRL, (c) Lienar-CAIRL, and (d) GAM-AIRL. We find (b) has two peaks when Vasopressors=2 and Fluids=3, which have higher dosages than what experts (a) do. It could be that the simulator requires higher dosages to have a difference in the predicted future states. And both (c) and (d) only prescribe fluids and ignores vassopressors which are dissimilar to (a) experts.

Ablation Study To determine the effectiveness of our design choices, we perform an ablation study in

Table 4: The ablation study. We show the test accuracy (%) in MIMIC3.

GAM-CAIRL	No BC regularization	No AIRL shaping term	No label smoothing	No input noise
74.7 ± 0.3	12.9 ± 3.0	74.5 ± 0.4	74.6 ± 0.3	74.4 ± 0.3

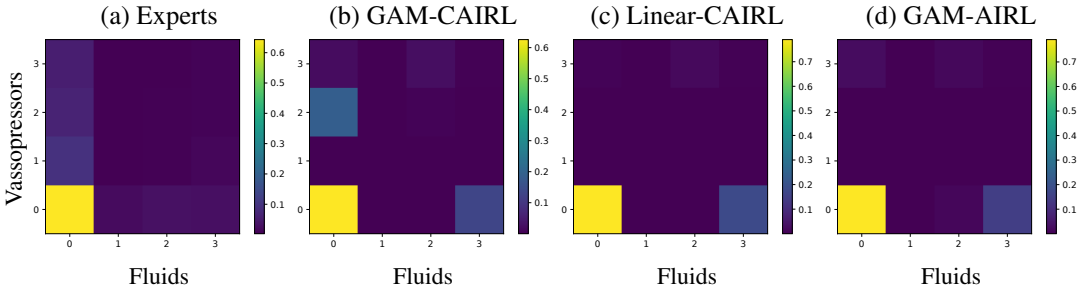

Figure 7: The action frequency of the (a) Experts, (b) GAM-CAIRL, (c) Linear-CAIRL, and (d) GAM-AIRL. GAM (b) prescribes both vasopressors and fluids similar to experts (a). But both (c) and (d) only prescribe fluids and ignore vassorpessors which is dissimilar to experts (a).

Table 4. The most effective component is the behavior cloning (BC) regularization where lack of it drastically reduces the accuracy. Including the AIRL shaping term and various tricks of stabilizing discriminators (label smoothing, input noise) slightly improve the performance. In Fig. 8, we conducted an sensitivity analysis under varying size of training data. Our method improves over the behavior cloning consistently with smaller training set.

6 Discussions, Limitations, and Conclusions

We emphasize that the test accuracy in batch IRL, unlike supervised learning, does not directly translate to how good the learned reward is. First, the policy can get good accuracy solely from the regularization such as the behavior cloning regularization and the early stopping. Also, the accuracy only tests the performance on the batch expert states but not its own induced states distributions which creates discrepancies to the online performance. As shown in our experiments, the incorrect form of linear rewards used in many prior works still gets quantitatively similar best accuracy (Table 3) but has wrong clinical reward (Fig. 5) and dissimilar action frequencies of the policy (Fig. 7). It shows the limitations of these offline metrics in real-world settings, and it's crucial to recover the reward in an interpretable way such as GAM that allows experts to audit and modify it.

We think that linear model generated the opposite reward is because when fitting non-linear data it compensates by changing the signs of correlated features thus creating counter-intuitive patterns.

We find GAM-CAIRL performs overall superior to GAM-AIRL - the accuracy is slightly higher (Table 3) , the recovered reward (Fig. 5) are better by recovering important guidelines while AIRL does not, and its action frequency is more similar to experts (Fig. 7). Notably, its accuracy gain is less significant as reported in CIRL (Bica et al., 2020) when comparing CIRL with MMA. It could be that our use of BC regularization reduce the accuracy gap, since the method would be at least as good as BC. Our dataset may also have higher correlation between the current and next states so AIRL which uses current states still operate well.

A limitation of our framework is that it needs additional complexity to estimate what the future looks like. Since we work with the batch data, we require assumptions about the dataset to make sure the future estimation is unbiased, and a strong violation of them may incur inaccurate estimation and subsequently wrong reward. Also, the temporal discretization needs to be done carefully. Vasopressors and fluids should take effect within 2 hours, thus we discretized our data in 2-hour windows. If treatments take longer to have an effect, a different discretization or considering multiple timesteps ahead might be needed. Finally, our method relies on the assumption that the reward is based on future outcomes. This may not be applicable in settings such as Altari game, but on many tasks in healthcare it is a reasonable representation of how humans operate.

In this work, we focus on an understudied literature that introduces the interpretability into the IRL. We propose to explain clinician's rewards by future outcomes and show our GAM explanations recovers clinical thresholds and patterns much better than the CIRL in a batch real-world clinical dataset. We also identify the failures of the linear model, and the pitfall of using offline metrics such as matched action accuracy, which are often used in prior works. We believe this interpretability is a critical step for IRL to be adopted in the high-stake, real clinical setting that helps better design human-aligned and safer rewards.

REPRODUCIBILITY STATEMENT

We provide our code anonymously in https://1drv.ms/u/s!ArHmmFHCSXTIhPVMTNlIkFDCO1VUXg?e=lNNb6o. We describe our MIMIC3 preprocessing in Supp. C, and our hyperparameters in Supp. F, Supp. G.1, and G.2. All experiments are run on 1 P100 GPU, 4 CPU and 16G RAM on a cluster.

REFERENCES

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 1, 2004.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Ioana Bica, Daniel Jarrett, Alihan Hüyük, and Mihaela van der Schaar. Learning" what-if" explanations for sequential decision-making. *arXiv preprint arXiv:2007.13531*, 2020.

Jessica Bunin. Diagnosis and management of hypotension and shock in the intensive care unit.

Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *KDD*, 2015.

Chun-Hao Chang, Mingjie Mai, and Anna Goldenberg. Dynamic measurement scheduling for event forecasting using deep rl. In *International Conference on Machine Learning*, pp. 951–960. PMLR, 2019.

Chun-Hao Chang, Sarah Tan, Ben Lengerich, Anna Goldenberg, and Rich Caruana. How interpretable and trustworthy are gams? In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 95–105, 2021.

Chun-Hao Chang, Rich Caruana, and Anna Goldenberg. NODE-GAM: Neural generalized additive model for interpretable deep learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=g8NJR6fCCl8.

Mehdi Fatemi, Taylor W Killian, Jayakumar Subramanian, and Marzyeh Ghassemi. Medical dead-ends and learning to identify high-risk states and treatments. *Advances in Neural Information Processing Systems*, 34, 2021.

Joseph Futoma, Michael C Hughes, and Finale Doshi-Velez. Popcorn: Partially observed prediction constrained reinforcement learning. *arXiv preprint arXiv:2001.04032*, 2020.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning*, pp. 1352–1361. PMLR, 2017.

Trevor Hastie and Robert Tibshirani. Generalized additive models for medical research. *Statistical methods in medical research*, 4(3):187–196, 1995.

Stefan Hegselmann, Thomas Volkert, Hendrik Ohlenburg, Antje Gottschalk, Martin Dugas, and Christian Ertmer. An evaluation of the doctor-interpretability of generalized additive models with interactions. In *Machine Learning for Healthcare Conference*, 2020.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29:4565–4573, 2016.

Simon Jenni and Paolo Favaro. On stabilizing generative adversarial training with noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12145–12153, 2019.

Fredrik Johansson, Uri Shalit, and David Sontag. Learning representations for counterfactual inference. In *International conference on machine learning*, pp. 3020–3029. PMLR, 2016.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

Alan E Jones, Lyn S Aborn, and Jeffrey A Kline. Severity of emergency department hypotension predicts adverse hospital outcome. *Shock*, 22(5):410–414, 2004.

Harmanpreet Kaur, Harsha Nori, Samuel Jenkins, Rich Caruana, Hanna Wallach, and Jennifer Wortman Vaughan. Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2020.

Ashish K Khanna. Defending a mean arterial pressure in the intensive care unit: Are we there yet? *Annals of intensive care*, 8(1):1–2, 2018.

Edouard Klein, Matthieu Geist, and Olivier Pietquin. Batch, off-policy and model-free apprenticeship learning. In *European Workshop on Reinforcement Learning*, pp. 285–296. Springer, 2011.

Brian K Lee, Justin Lessler, and Elizabeth A Stuart. Improving propensity score weighting using machine learning. *Statistics in medicine*, 29(3):337–346, 2010.

Donghun Lee, Srivatsan Srinivasan, and Finale Doshi-Velez. Truly batch apprenticeship learning with deep successor features. *arXiv preprint arXiv:1903.10077*, 2019.

Bryan Lim, Ahmed M Alaa, and Mihaela van der Schaar. Forecasting treatment responses over time using recurrent marginal structural networks. *NeurIPS*, 18:7483–7493, 2018.

Michael Oberst and David Sontag. Counterfactual off-policy evaluation with gumbel-max structural causal models. In *International Conference on Machine Learning*, pp. 4881–4890. PMLR, 2019.

Ahmed H Qureshi, Byron Boots, and Michael C Yip. Adversarial imitation via variational inverse reinforcement learning. *arXiv preprint arXiv:1809.06404*, 2018.

James M Robins, Miguel Angel Hernan, and Babette Brumback. Marginal structural models and causal inference in epidemiology, 2000.

Paul R Rosenbaum and Donald B Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29: 2234–2242, 2016.

Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Peter Schulam and Suchi Saria. Reliable decision support using counterfactual models. *Advances in Neural Information Processing Systems*, 30:1697–1708, 2017.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Srivatsan Srinivasan and Finale Doshi-Velez. Interpretable batch irl to extract clinician goals in icu hypotension management. *AMIA Summits on Translational Science Proceedings*, 2020:636, 2020.

Elizabeth A Stuart. Matching methods for causal inference: A review and a look forward. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 25(1):1, 2010.

Sarah Tan, Rich Caruana, Giles Hooker, and Yin Lou. Distill-and-compare: Auditing black-box models using transparent model distillation. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 303–310, 2018.

Sarah Tan, Rich Caruana, Giles Hooker, Paul Koch, and Albert Gordo. Learning global additive explanations of black-box models. 2019.

Yongxin Yang, Irene Garcia Morillo, and Timothy M Hospedales. Deep neural decision trees. *arXiv preprint arXiv:1806.06988*, 2018.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.

A CAUSAL ASSUMPTIONS

Here we illustrate the causal assumptions we make in order to learn the correct potential outcomes under observstional patient data for the counterfactual transition model. Why do we need these assumptions? This is because we have only access to a batch clinical dataset (e.g. electronic health records) in which the outcomes of patients are entangled with the treatments they receive. For example, seriously ill patients are more likely to get aggressive drug treatments, but they are also more likely to have adverse outcomes. A model without corrections might wrongly conclude that the drug treatments lead to adverse outcomes.

To identify the counterfactual outcomes from observational data we make the standard assumptions of consistency, positivity and no hidden confounders as described in Assumption 1 (Rosenbaum & Rubin, 1983; Robins et al., 2000; Bica et al., 2020). Under Assumption 1, we can estimate potential outcome $E[Y_{t+1}[a_t]|h_t] = E[X_{t+1}|a_t, h_t]$ by training a regression model on the batch observational data.

Assumption 1 (Consistency, Ignorability and Overlap). For any individual i, if action a_t is taken at time t, we observe $X_{t+1} = Y_{t+1}[a_t]$. Moreover, we have sequential strong ignorability (no hidden confounder) assumption that $\{Y_{t+1}[a]_{a\in A}\} \perp a_t | h_t$ for any t, and sequential overlap $\Pr(A_t = a|h_t) > 0$ for all a, t.

We expand upon what these assumptions mean in practice as follows. Consistency means that the actions recorded in the data are in fact performed and change the outcome. For example, it could be that some patients are assigned drugs but never take them because they don't believe in them. Positivity means each state-action pair has a non-zero probability of happening. For example, if male patients always get treated, we would not know what would happen if the male patients do not get treated and thus can not estimate the corresponding state. It is difficult to verify this requirement in the continuous space. Finally, No Hidden Confounder means there is no unknown factor that both changes the treatments and the outcomes. For example, wealth can be a hidden confounder if we believe wealth changes how doctors treat patients (e.g. giving higher-price treatments) and wealth changes outcomes (wealthier patients are healthier), then wealth could be a hidden confounder if it is not used when constructing the state space. In practice, there's often no easy way to check these assumptions and it requires experts to carefully analyze the data.

These assumptions are standard across causal inference methods (Schulam & Saria, 2017; Lim et al., 2018). If these assumptions are valid, we can learn an unbiased model by either matching (Stuart, 2010), propensity weighting (IPTW, Lee et al. (2010)), or adversarial representation learning (Johansson et al., 2016). In this paper, we follow the time-series version of propensity weighting (Lim et al., 2018) to learn an unbiased model.

B THE BACKGROUND OF AIRL (QURESHI ET AL., 2018)

For completeness we describe the AIRL here and refer the readers to Qureshi et al. (2018) for more details. The AIRL builds upon the Maximum Causal Entropy IRL (Ziebart et al., 2008), which finds a cost function c (the negative of the reward) from a family of functions C which has both high

entropy and minimum cumulative cost (i.e. maximum reward). Given the cost c, policy π, expert policy π_E, entropy H, it optimizes:

$$\max_{c \in C} (\min_{\pi \in \Pi} -H(\pi) + \mathbb{E}_{\pi}[c(s,a)]) - \mathbb{E}_{\pi_E}[c(s,a)]$$

Maximum Casual entropy IRL maps a cost function to high-entropy policies that minimize the expected cumulative cost (or maximize the cumulative reward). GAIL (Ho & Ermon, 2016) further introduces a convex regularization cost ψ to limit the space of cost function c, so the optimization becomes:

$$\text{IRL}_{\psi}(\pi_E) = \arg\max_{c \in C} -\psi(c) + (\min_{\pi \in \Pi} -H(\pi) + \mathbb{E}_{\pi}[c(s,a)]) - \mathbb{E}_{\pi_E}[c(s,a)].$$

And it can be shown that optimizing a RL policy under the recovered cost from IRL is equivalent to match the occupancy measure to the experts' occupancy, as measured by the dual ψ^*:

$$\text{RL} \circ \text{IRL}(\pi_E) = \arg\min_{\pi \in \Pi} -H(\pi) + \psi^*(\rho_{\pi} - \rho_{\pi_E})$$

Here, the occupancy measure (ρ) is the distribution of state-action pair encountered while navigating the environment with the policy. Given the above result, we can look at IRL from a different view as a procedure that tries to induce a policy that matches the expert's occupancy measure. Therefore, GAIL proposes to optimize it following the adversarial framework that the discriminator tries to differentiate between expert occupancy and the induced policy occupancy, while the generator (policy) aims to cheating the discriminator while maximizing the entropy. If we choose ψ^* as the Jenson-Shenon Divergence D_{js}, then we optimize the discriminator D as:

$$\max_{D} \mathbb{E}_{\pi}[\log(D(s,a))] + \mathbb{E}_{\pi_E}[\log(1 - D(s,a))].$$

However, AIRL (Qureshi et al., 2018) finds GAIL (Ho & Ermon, 2016) does not recover the expert reward, and instead they recover the reward $f(s,a)$ by formulating the output of the discriminator as:

$$D_{\theta}(s,a) = \frac{\exp(f_{\theta}(s,a))}{\exp(f_{\theta}(s,a)) + \pi(a|s)}.$$

Intuitively, if the states are from experts where the $\pi(a|s)$ is close to 0, the reward f should increase to make D close to 1. On the other hand, if the states are from policy π with $\pi(a|s)$ close to 1, the reward f should decrease that make the D close to 0.

Moreover, since the reward can have a set of transformations that induces the same optimal policy, called **reward shaping**. Given the transformed reward r', current state s, action a, next state s', discount factor γ, and transformations Φ,

$$r'(s,a,s') = r(s,a,s') + \gamma\Phi(s') - \Phi(s).$$

It's because the above Φ canceled out under cumulative reward in a trajectory. To solve it, AIRL proposes to decompose the reward f into a reward term g and a shaping term h:

$$f(s,a,s') = g(s,a) + \gamma h(s') - h(s).$$

And g recovers the ground truth reward r^* up to a constant C:

$$g(s,a) = r^*(s,a) + C.$$

The Differences between CAIRL and AIRL Our CAIRL modifies two assumptions that AIRL makes. First, instead of assuming reward r comes from the current state s, CAIRL assumes the reward depending on the future states s' to better resembles the clincians' reasoning. Second, we model the reward g using an interpretable GAM model parameterized on future states s'. This gives us an explainable rewards that allows clinicians to verify if the recovered reward is valid.

C MIMIC3 PREPROCESSING

We follow Futoma et al. (2020) to extract 5 covariates, 29 time-varying features and 10 features related to actions. We use the quantile transformation to Gaussian distribution and finds models

provide more meaningful results than the log transformation used in Futoma et al. (2020). For action features, we calculate the past treatment values including the treatment in the last time point (e.g. last_fluid_2 means if the patient gets treated in the last time point with value 2 (medium)), and the treatment value in the last 8 hours, and the total treatment values so far. We also include the missingness indicator for each feature since medical data is not missing at random which results in total 73 features.

- covariates: age, is_F, surg_ICU, is_not_white, is_emergency, is_urgent
- features: dbp, fio2, hr, map, sbp, spontaneousrr, spo2, temp, urine, weight, bun, magnesium, platelets, sodium, alt, hct, po2, ast, potassium, wbc, bicarbonate, creatinine, lactate, pco2, bilirubin_total, glucose, inr, hgb, GCS
- action features: last_vaso_1, last_vaso_2, last_vaso_3, last_fluid_1, last_fluid_2, last_fluid_3, total_all_prev_vasos, total_all_prev_fluids, total_last_8hrs_vasos, total_last_8hrs_fluids

D THE DESCRIPTIONS OF THE FIG. 5 (I)-(P)

In the third row (i)-(l), urine volume (i) is correlated with blood pressure and usually high output is a good sign of health while low volume (¡50) is an indicator of acute or chronic kidney disease. GAM successfully captures this trend by learning much lower reward for low urine especially below 50, while Linear model learns a higher reward for lower urine output. Body temperature (j) should be maintained between 36-38 degrees. Values that are higher or lower are concerning; GAM captures it perfectly while Linear model learns the upward trend that higher the temperature the better, failing to capture the needed non-linearity. Notably, GAM-AIRL here also fails to show a lower reward for higher or lower temperature. WBC (k) has a normal range between $4.5 - 11$, and high WBC often indicates an infection. GAM displays a steady decrease once the threshold of 10 is exceeded. FiO2 (l) is usually maintained below 50 even when ventilation is used to avoid oxygen toxicity, and we clearly see a sharp decrease of reward above 50 in GAM, but Linear model again learns a counter-intuitive trend. GAM-AIRL also fails to learn this sharp decrease and mostly remains flat.

In the last row (m)-(p), ALT (m) is an enzyme found in the liver, and a high ALT value implies damaged liver that releases ALT into the bloodstream. GAM captures this and prefers the lower value and quickly decreases reward when value exceeds 50, matching the clinical guideline. And again Linear model learns the opposite. INR (n) is a prothrombin time (PT) test that measures the time it takes for the liquid portion of one's blood to clot, and normal people have values below 1.1. GAM captures this threshold by modeling a sharp decrease of the reward after 1.1, while Linear model is unable to learn this threshold effect. BUN (o) measures the amount of urea nitrogen in the blood, and a high BUN level implies worse conditions like heart failure or shock. Both GAM and Linear model capture the correct downturn trend but GAM learns a sharp drop around 8 similar to the clinical guideline. Finally, Bilirubin (p) is a yellow pigment that occurs normally when part of one's red blood cells break down. High bilirubin levels are a sign that the liver isn't clearing the bilirubin from one's blood as it should. GAM again captures the important clinical threshold of 1.2 while the Linear model has the opposite trend learning that higher BUN is better.

E SEPSIS SIMULATION REWARD DESIGN

In GAM MDP, we simulate the reward using the following state value pair:

- Heart rate: {0: -0.8, 1: 0, 2: -1}
- Systolic BP: {0: -1.2, 1: 0, 2: -0.6}
- % of Oxygen: {0: -1, 1: 0}
- Glucose: {0: -0.8, 1: -0.4, 2: 0, 3: -0.4, 4: -0.8}

In Linear MDP, we simulate the reward using the following state value pair:

- Heart rate: {0: -0.3, 1: -0.6, 2: -0.9}
- Systolic BP: {0: -0.4, 1: -0.8, 2: -1.2}

Table 5: Hyperparameters for GRU training for Behavior Cloning (BC) and Transition Model. We use random search to find the best hyperparameters.

	GRU BC	GRU Transition Model
Epochs	200	200
Batch size	64, 128, 256	128, 256
Learning Rate	5e-4, 1e-3, 2e-3	5e-4, 1e-3
Weight Decay	0, 1e-6, 1e-5, 1e-4	0, 1e-5
GRU Num Hidden	64, 128, 256	64
GRU Num layers	1	1
GRU Dropout	0.3, 0.5	0.3, 0.5
FC Num Hidden	128, 256, 384, 512	256, 384, 512
FC Num Layers	2, 3, 4	2
FC Dropout	0.15, 0.3, 0.5	0.15
FC Activation	ELU	ELU
Act Num Hidden	-	64, 128
Act Num Layers	-	0, 1, 2
Act Num output	-	32, 64, 96
Act Dropout	-	0.3

- % of Oxygen: $\{0: 0, 1: 0.6\}$
- Glucose: $\{0: 0, 1: 0.2, 2: 0.4, 3: 0.6, 4: 0.8\}$

F GRU TRAINING FOR BEHAVIOR CLONING AND COUNTERFACTUAL TRANSITION MODEL

Behavior Cloning (BC) Behavior cloning model takes in the history h_t to predict the current action a_t in the expert batch data. We use GRU to model the prediction. So we feed the history h_t into the GRU to produce output o_t, and then feed the output into several layers of fully-connected layers (FC) with dropout and batchnorm to produce the final classification of action a_t. We list the hyperparameters in Table 5.

Counterfactual Transition Model Learning Counterfactual models take in the current history h_t and action a_t to predict the next states s_{t+1}. We use the similar architecture as the GRU for behavior cloning, except we also use action a_t as inputs and do the regression to predict s_t. Specifically, for action a_t, we go through several layers of Fully Connected Layers to produce action embeddings ϕ_a, and concatenate with the state s_t as the inputs to the GRU model. For output, we use the Huber loss (smoothed ℓ_1 loss) that we find it produces more diverse states than Mean Squared Error (MSE) loss. Finally, we weight the samples by stabilized Inverse Propensity Weighting (IPTW) that gives different sample weights w_t. We list the hyperparameters in Table 5. Specifically, given the behavior cloning policy π_{bc}, action embedding layers (Act) A, GRU model G and output fully connected layer F, we have

$$
\begin{aligned}
\phi_a &= A(a_t) \\
\phi &= \text{concat}(s_t, \phi_a) \\
g_t &= M(\phi) \\
o_t &= F(g_t) \\
w_t &= \frac{P(a_t)}{\pi_{bc}(a_t)} \\
L &= w_t \cdot \text{Huber}(o_t, s_{t+1}) \\
\theta &\longleftarrow \theta - \nabla_\theta L \quad \text{(Updated by Adam)}
\end{aligned}
$$

Table 6: Hyperparameters for Node-GAM training for discriminator training. We use random search to find the best hyperparameters.

	Simulation	MIMIC3
Epochs	100	100
Input Noise	0	0,0.1
Noise decay	0	80%
LR	2e-4, 4e-4	5e-4, 8e-4, 1e-3
Label Smoothing δ	0	0,0.005,0.01
Num Layers	1, 2	1,2,3
Num Trees	100, 200, 400	200,300,400
Addi Tree Dim	0, 1	0, 1
Depth	1, 2	2,3,4
Output Dropout	0, 0.1	0.1,0.2
Last Dropout	0, 0.3	0.3,0.5
Column Subsample	1, 0.5	0.5
Temp Annealing	3000	3000

G AIRL TRAINING

G.1 DISCRIMINATOR TRAINING

Here we train a discriminator that can distinguish expert batch data as class 1 and generated experience as class 0 in the binary classification setting. And its logit would represent the expert reward r. Given a batch of expert data X_E, we generate the data X_G by executing the generator policy π in the trained counterfactual transition model T. Note that to explain the expert in terms of future counterfactuals, we exclude the static covariates and action features and only use the time-varying features of next state when training the discriminator.

Discriminator Stablizing Tricks When optimizing discriminator, we use both one-sided label smoothing (Salimans et al., 2016) which reduces the label confidence for the expert batch data. We also add a small input Gaussian noise to the inputs for both expert and generated data, and linearly decayed the noise throughout the training. It has been shown useful to stabilize GAN adversarial optimization (Jenni & Favaro, 2019). Specifically, given label smoothing δ, input noise δ_n, the discriminator model $\mathcal{D}$, expert batch data D_E, the transition model T, the generator policy π_G and binary cross entropy loss (BCE):

$$
\begin{aligned}
&X_E = (s, a, s') \sim D_E \\
&X_G = (s, a, s') \text{ where } s \sim X_E, a \sim \pi_G(\cdot|s) \text{ and } s' \sim T(\cdot|s, a) \\
&n \sim \mathcal{N}(\mathbf{0}, \mathbf{1}) \\
&X_E = X_E + \delta_n \cdot n,\ X_G = X_G + \delta_n \cdot n \\
&L = \text{BCELoss}(D(X_E), \mathbf{1} - \delta) + \text{BCELoss}(D(X_G), \mathbf{0})
\end{aligned}
$$

We list the hyperparameters we use for Node-GAM in Table 6. And we list the linear and FCNN model's hyperparameters in Table 7.

G.2 GENERATOR TRAINING

In Sepsis simulation, since we use the value iteration to solve the exact optimal policy for our generator, there is no hyperparameter to tune. To save the computation, we only update the generator after the discriminator updates for 20 steps.

In MIMIC3, we use the soft-Q learning as our generator as described in Sec. 4. We use a fully connected neural net with dropout, ELU activation function and batchnorm as our architecture. We show the hyperparameters in Table 8.

Table 7: Hyperparameters for Linear model and FCNN model for discriminator training.

	Linear		FCNN	
	Simulation	MIMIC3	Simulation	MIMIC3
Epochs	100	100	100	100
Input Noise	0	0,0.1	0	0,0.1
Noise decay	0	0.8	0	0.8
LR	2e-4, 4e-4	5e-4, 8e-4, 1e-3	2e-4, 4e-4	5e-4, 8e-4, 1e-3
Label Smoothing δ	0	0,0.005,0.01	0	0,0.005,0.01
Num Layers	-	-	2,3,4,5	2,3,4,5
Num Hidden	-	-	32,64,128,256	32,64,128,256
Dropout	-	-	0.1,0.3,0.5	0.1,0.3,0.5

Table 8: Hyperparameters for generator training in Sepsis and MIMIC3 datasets.

	Sepsis	MIMIC3
Update Freq	20	1
Epochs	-	100
Entropy Coeff α	-	0.25, 0.5
Sample weights for gen data (δ)	-	0.5
Sync Rate	-	200
LR	-	4e-4, 8e-4
Num Layer	-	3, 4
Dropout	-	0.3,0.5
BC Reg	-	10
BC Reg decay	-	0.5

H REWARD SCALING

Since the reward can be arbitrarily shifted and scaled without changing the resulting optimal policy, comparing the reward across models requires setting the scale of the reward for each model when showing the GAM plots and calculating distance. Therefore, in the simulation for each model, we shift the average reward to 0 and set the scaling a that has the smallest ℓ_1 distance to the ground truth reward under the state distribution of the expert batch data. Given the ground truth model G and its GAM main effect $f_G(x_j)$ of each feature j, model M and $f_M(x_j)$, V_j as all the values of feature j, with each value $v \in V_j$, and the counts $c(v)$ in the expert batch data, we derive a by convex optimization:

$$\min_a \sum_{j=1}^{D} \sum_{v \in V_j} |(f_G(v) - a f_M(v))| c(v)$$

In real-world data where there is no ground truth, we choose the scale a to minimize the difference of max and min value in each feature of two models G, M to make them display in the similar range:

$$\min_a \sum_{j=1}^{D} (|\min_{v \in V_{Gj}} f_G(v) - a \min_{v \in V_{Mj}} f_M(v)| + |\max_{v \in V_{Gj}} f_G(v) - a \max_{v \in V_{Mj}} f_M(v)|).$$

I CLINICIAL GUIDELINES SOURCES

Here we list the lower and upper bound, and the sources of the normal ranges we use in Fig. 5.

- MAP: (70, 100) `https://www.healthline.com/health/mean-arterial-pressure#:~:text=What%20is%20a%20normal%20MAP,100%20mmHg%20to%20be%20normal.`

- Lactate: (0, 2)
- Systolic BP: upper bound 180 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3704960/ and lower bound 90 https://www.mayoclinic.org/diseases-conditions/low-blood-pressure/symptoms-causes/syc-20355465#:~:text=What's%20considered%20low%20blood%20pressure,pressure%20is%20lower%20than%20normal..
- Bicarbonate: (23, 30) https://www.urmc.rochester.edu/encyclopedia/content.aspx?contenttypeid=167&contentid=bicarbonate#:~:text=Normal%20bicarbonate%20levels%20are%3A,30%20mEq%2FL%20in%20adults
- pO2: (75, 100) https://www.medicalnewstoday.com/articles/322343#:~:text=Most%20healthy%20adults%20have%20a,emphysema
- Heart Rate: (40, 100) https://health.clevelandclinic.org/is-a-slow-heart-rate-good-or-bad-for-you/
- Potassium: (2.5, 5.1) https://my.clevelandclinic.org/health/diseases/17740-low-potassium-levels-in-your-blood-hypokalemia
- HCT: (35.5, 48.6) https://www.mayoclinic.org/tests-procedures/hematocrit/about/pac-20384728
- Urine: (400, inf) https://www.healthline.com/health/urine-output-decreased#:~:text=Oliguria%20is%20considered%20to%20be,is%20considered%20to%20be%20anuria.
- WBC: (4.5, 11) https://my.clevelandclinic.org/health/diagnostics/17704-high-white-blood-cell-count
- FiO2: (21, 50) https://en.wikipedia.org/wiki/Fraction_of_inspired_oxygen#:~:text=Natural%20air%20includes%2021%25%20oxygen,to%20100%25%20is%20routinely%20used.
- ALT: (0, 55) https://www.mayoclinic.org/tests-procedures/liver-function-tests/about/pac-20394595
- INR: (0, 1.1) https://my.clevelandclinic.org/health/diagnostics/17691-prothrombin-time-pt-test
- BUN: (2.1, 8.5) https://www.mayoclinic.org/tests-procedures/blood-urea-nitrogen/about/pac-20384821
- Bilirubin total: (0, 1.2) https://www.webmd.com/a-to-z-guides/bilirubin-test

J Computing Resources Used

All experiments are run on 1 P100 GPU, 4 CPU and 16G RAM on a cluster.

K Ablation study: performance under various training sample sizes

To analyze if our performance still improves over various number of training size, we conducted an sensitivity analysis under varying number of training data in Fig. 8. We find that our method improves over the behavior cloning consistently even when training set is small.

L Complete shape graphs

We show the complete shape graphs of 29 features in MIMIC-III in Fig. 9.

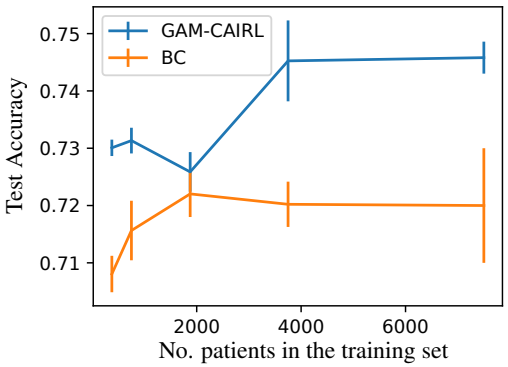

Figure 8: The test accuracy under varying training samples in MIMIC3. We find CAIRL's performance outperforms behavior cloning (BC) consistently.

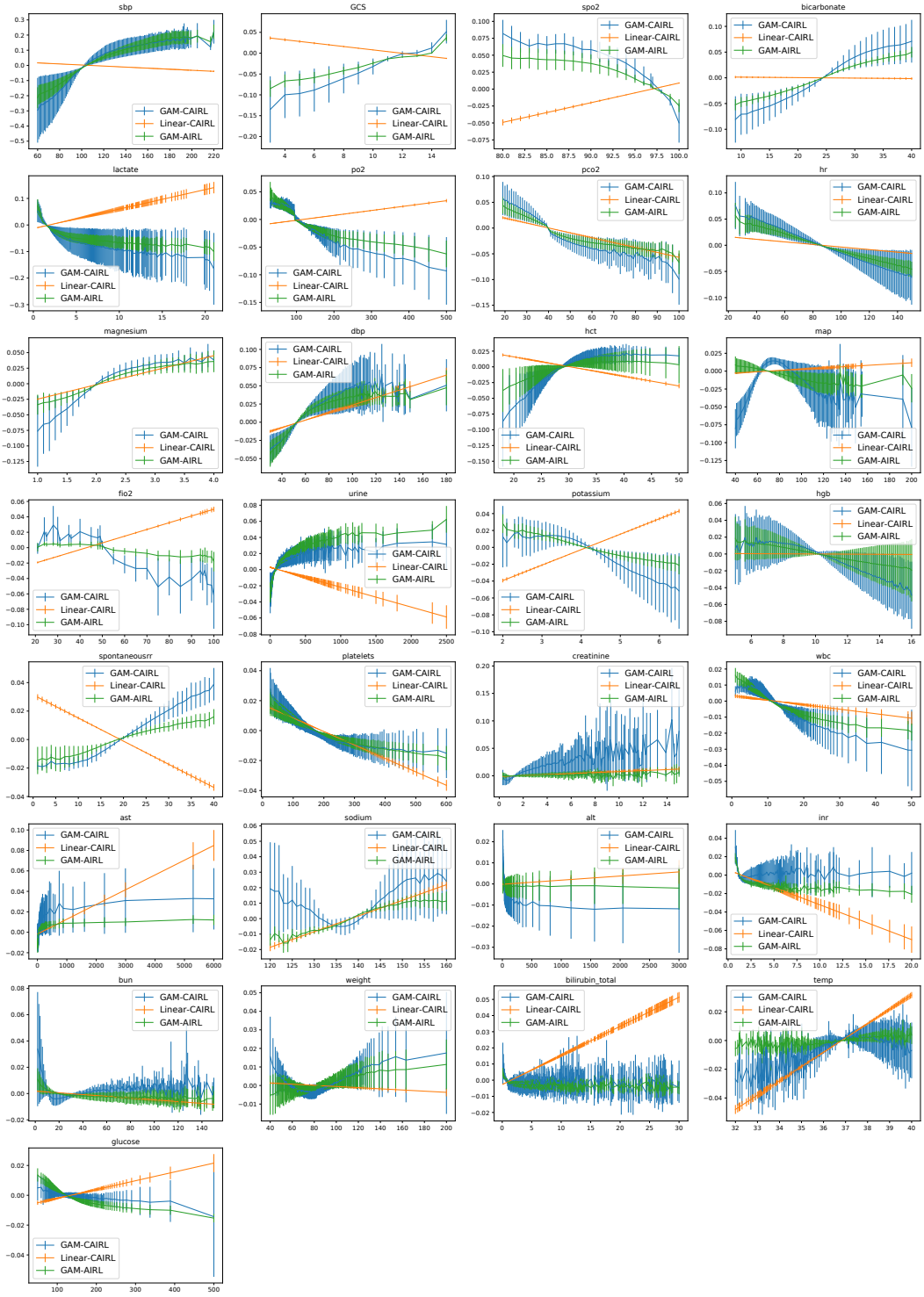

Figure 9: The complete shape plots of MIMIC3.