REINFORCEMENT LEARNING WITH EFFICIENT ACTIVE FEATURE ACQUISITION

Anonymous authors
Paper under double-blind review

ABSTRACT

Solving real-life sequential decision making problems under partial observability involves an exploration-exploitation problem. To be successful, an agent needs to efficiently gather valuable information about the state of the world for making rewarding decisions. However, in real-life, acquiring valuable information is often highly costly, e.g., in the medical domain, information acquisition might correspond to performing a medical test on a patient. This poses a significant challenge for the agent to perform optimally for the task. In this paper, we propose a model-based reinforcement learning framework that learns a policy which solves this exploration-exploitation problem during its execution. Key to the success is a novel sequential variational auto-encoder that learns high-quality representations from partially observed states, which are then used by the policy to maximize the task reward in a cost efficient manner. We demonstrate the efficacy of our proposed framework in a control domain as well as using a medical simulator. In both tasks, our proposed method outperforms conventional baselines and results in policies with greater cost efficiency.

1 INTRODUCTION

Recently, machine learning models for automated sequential decision making have shown remarkable success across many application areas, such as visual recognition (Mathe et al., 2016; Das et al., 2017), robotics control (Finn et al., 2016; Zhang et al., 2018), medical diagnosis (Ling et al., 2017; Peng et al., 2018) and computer games (Mnih et al., 2015; Silver et al., 2016). One fundamental reason that drives the success of such models and enables them to outperform classical algorithms is the availability of large amounts of training data. Typically this training data is either fully observed or the features stem from an action-independent observation model (which clearly can depend on the state of the system). However, the fundamental assumption that the same features are readily available during deployment does not hold in many real-world applications. For instance, consider a medical support system for monitoring and treating patients during their stay at hospital which was trained on rich historical medical data. To provide the best possible treatment, the system might need to perform several measurements of the patient over time, and some of them could be costly or pose a health risk. That is, at the deployment the system should function with minimal features while during training more features might have been available. Hence, in such cases, we are interested in decision making models that take the measurement process, i.e., the feature acquisition, into account and only acquire the information relevant for making a decision.

In this paper, we consider the challenging problem of learning effective policies when the cost of information acquisition cannot be neglected. To be successful,we need to learn policies which acquires the information required for solving a task in the cheapest possible way. For simplicity, we can think of the policy as being constituted of an *acquisition policy* which selects the features to be observed and a *task policy*, which selects actions to change the state of the system towards some goal.[1] As such, we consider a partially observable learning problem with the following two distinguishing properties compared to the most commonly studied problems (see also Figure 3.2 for an illustration). *(i)* By incorporating active feature acquisition, the training of the task policy is based upon subsets of features only, i.e., there are missing features, where the missingness is

[1]Clearly, these two policies are not independent in general, e.g., acquiring features can change the state of the system.

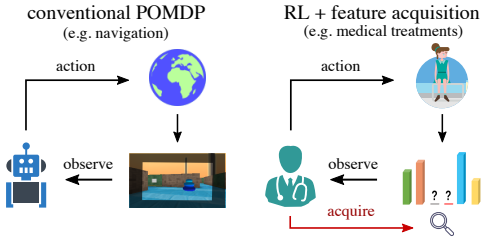


Figure 1: An illustrative figure for comparing POMDPs considered in conventional RL literature and those in our proposed setting. By performing active feature acquisition, the partial observability is determined by the feature acquisition policy besides the evolvement of the system.

controlled by the acquisition policy. Thus, the resulting POMDP is different from typically considered POMDPs in RL literature (Cassandra, 1998) where the partial observability stems from a fixed and action-independent observation model. Also, the state-transitions in conventional POMDPs are often only determined by the choice of the task action, whereas in our setting the state-transition is affected by both the task action and the feature acquisition choice. *(ii)* The learning of the acquisition policy introduces an additional dimension to the explore-exploit problem: each execution of the policy needs to solve an explore-exploit problem, and thus we often need to learn sophisticated policies.

Most reinforcement learning research has not taken active feature acquisition into consideration. In this work, we propose a unified approach that jointly learns a policy for optimizing the task reward while performing active feature acquisition. Although some of the prior works have exploited the use of reinforcement learning for sequential feature acquisition tasks (Shim et al., 2018; Zannone et al., 2019), they considered variable-wise information acquisition in a static setting only, corresponding to feature selection for non-time-dependent prediction tasks. However, our considered setting is truly time-dependent and feature acquisitions need to be made at each time step while the state of the system evolves simultaneously. As such, both the model dynamics of the underlying MDP and the choice of feature acquisition introduce considerable challenges to learning the sequential feature acquisition strategy.

Due to the challenge of the exploration-exploitation problem, it is a non-trivial task to jointly learn the policies. The conventional end-to-end approaches often result in inferior solutions in complex scenarios. Ideally, policies based on high-quality representations would be easier for the algorithm to search for better solutions through exploration-exploitation. Therefore, our presented framework also tackles the joint policy training task from a representation learning perspective. Specifically, we introduce a novel representation learning method that not only encodes the sequential partially observed information into its latent features, but also efficiently imputes the unobserved features to offer more meaningful information for the policy training. To this end, we present a sequential generative model that can efficiently learn model dynamics during representation learning. Overall, our contributions are three-fold:

- We propose an approach for learning policies with active feature acquisition through reinforcement learning. Our proposed approach simultaneously learns policies for reward optimization and active feature acquisition.
- We present a novel sequential representation learning approach to account for the encoding of the partially observed states. Our proposed approach is based on variational autoencoders (VAE) with amortized inference. The imputation of the unobserved features is achieved via learning model dynamics.
- We demonstrate that our presented framework can be applied to various applications. We conduct extensive experiments on an image-based control task and a medical simulator fitted from real-life data. Our method shows clear improvements over conventional baselines.

2 Related Work

In this work, we integrate active learning with reinforcement learning to accomplish the policy training task while acquiring fewer observed features as possible. We thus review related methods for active feature acquisition and representation learning for POMDP, respectively.

2.1 ACTIVE FEATURE ACQUISITION

Our work draws motivation from existing instance-wise active feature selection approaches. One category of the instance-wise feature selection methods consider feature acquisition as a one time effort to select a subset of features as a whole. One typical example is the conventional linear model that poses sparsity inducing prior distribution to the model (Tibshirani, 1996). Recently, there also emerged approaches that adopt reinforcement learning to actively find optimal feature subsets (Yoon et al., 2018; Zannone et al., 2019; Shim et al., 2018). Though such attempts have demonstrated certain efficacy in handling non time-series instance-wise data, they do not suffice for handling sequential dataset. There is an alternative category that models feature acquisition as a Bayesian experiment design (Ma et al., 2019; Gong et al., 2019). However, the sequential decision making is for variable-wise feature acquisition and the problems are still non time-series tasks in nature.

The key difference between all the existing approaches and ours is in that we tackle active feature acquisition for the time-series data, where an active feature selection decision needs to be formed at each time step along the multi-step reinforcement learning trajectory. Therefore, the feature acquisition for our presented work needs to consider more complex information over model dynamics and control, apart from static instance-wise features.

2.2 REPRESENTATION LEARNING IN POMDP

In complex tasks, policies trained upon different representations can even converge to different performance levels. Most conventional (deep) reinforcement learning approaches unifies the process of representation learning with policy training and results in policies trained in an end-to-end fashion (Mnih et al., 2013; Lillicrap et al., 2016; Mnih et al., 2016). However, to accomplish the representation learning task, such models often introduce additional training parameters which could come with considerable size and thereby result in significant sample inefficiency.

When considering problems with POMDPs where the state space are partially presented to the agent, representation learning becomes an important and non-trivial research challenge. Among the existing literature, one prominent line of research tackles the representation learning for POMDP in an off-line fashion and thus resulting in multi-stage reinforcement learning. Higgins et al. (2016; 2017) adopt pretrained VAE models as representation module to build agents with strong domain adaptation performance. The key difference between their works and ours is in that they encode instance-wise image frames from POMDP domains where the image presents partial view over the task environment, our work considers the setting with a distinct partial observability for cost-sensitive reinforcement learning problems, i.e., feature-level information is *missing* at each time step. We thus adopt a sequential representation learning approach to infer a more representative state information. Recently, there emerged a fruitful literature over sequential representation learning for POMDP (Gregor et al., 2019; Vezzani et al., 2019). However, most of the works utilize VAE training as an auxiliary task to jointly update the representation model with the policy. In our work, due to the high acquisition cost to the entire feature set, we consider off-line representation learning. Moreover, we introduce a novel model-based representation learning mechanism, where imputing the missing features brings significant benefit to derive high-quality representation for policy training.

3 METHODOLOGY

3.1 TASK SETTING

In this section, we formally define the problem settings for the task of jointly learning the task and feature acquisition policy. To this end, we define the *active feature acquisition POMDP*, a rich class of discrete-time stochastic control processes generalizing standard POMDPs:

Definition 1 (AFA-POMDP)**.** *The* active feature acquisition POMDP *is a tuple* $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{R}, \mathcal{C}, \gamma \rangle$*, where* $\mathcal{S}$ *is the state space and* $\mathcal{A} = (\mathcal{A}^f, \mathcal{A}^c)$ *is a joint action space of feature acquisition actions* $\mathcal{A}^f$ *and control actions* $\mathcal{A}^c$*. The transition kernel* $\mathcal{T} \colon \mathcal{S} \times \mathcal{A}^c \times \mathcal{A}^f \to P_{\mathcal{S}}$ *maps any joint action* $\mathbf{a} = (\mathbf{a}^f, \mathbf{a}^c)$ *in state* $s \in \mathcal{S}$ *to a distribution* $P_{\mathcal{S}}$ *over next states. In each state* s*, when taking action* $\mathbf{a}^f$*, the agent observes* $\mathbf{x}^p = \mathbf{x}(\mathbf{a}^f)$*, i.e., a subset of the features* $\mathbf{x} = (\mathbf{x}^p, \mathbf{x}^u) \sim \mathcal{O}(s)$ *indicated by* $\mathbf{a}^f$*, where* $\mathcal{O}(s)$ *is a distribution over possible feature observation for state* s *and* $\mathbf{x}^u$ *are features not observed by the agent. When taking a joint action, the agent obtains*

rewards according to the reward function $\mathcal{R}\colon \mathcal{S} \times \mathcal{A}^c \to \mathbb{R}$ *and pays a cost of* $\mathcal{C}\colon \mathcal{S} \times \mathcal{A}^f \to \mathbb{R}_+$ *for feature acquisition. Rewards and costs are discounted by the discount factor* $\gamma \in [0, 1)$.

Simplifying assumptions For simplicity, we assume that $\mathbf{x}$ consists of a fixed number of features N_f for all states, that $\mathcal{A}^f = 2^{[N_f]}$ is the powerset of all the N_f features, and that $\mathbf{x}^p(\mathbf{a}^f)$ consists of all the features in $\mathbf{x}$ indicated by the subset $\mathbf{a}^f \in \mathcal{A}^f$. Note that the feature acquisition action for a specific application can take various different forms. For instance, in our experiments in Section 4, for the *Sepsis* task, we define feature acquisition as selecting a subset over possible *measurement* tests, whereas for the *Bouncing Ball*$^+$ task, we divide an image into four observation regions and let the feature acquisition policy select a subset of observation regions (rather than raw pixels). Please also note that while in a general AFA-POMDP, the transition between two states depends on the joint action, we assume in the following that it depends only on the control action, i.e., $\mathcal{T}(s, \mathbf{a}^c, \mathbf{a}^{f'}) = \mathcal{T}(s, \mathbf{a}^c, \mathbf{a}^f)$ for all $\mathbf{a}^{f'}, \mathbf{a}^f \in \mathcal{A}^f$. While not true for all possible applications, this assumption can be a reasonable approximation for instance for medical settings in which tests are non-invasive. For simplicity we furthermore assume that acquiring each feature has the same cost, denoted as c, i.e., $\mathcal{C}(\mathbf{a}^f, s) = c\,|\mathbf{a}^f|$, but our approach can be straightforwardly adapted to have different costs for different feature acquisitions.

Objective We aim to learn a policy which trades off reward maximiziation and the cost for feature acquisition by jointly optimizing a task policy π^c and a feature acquisition policy π^f. That is, we aim to solve the optimization problem

$$\max_{\pi^f, \pi^c} \quad \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \Big(\mathcal{R}(\mathbf{x}_t, \mathbf{a}_t^c) - \sum_{i}^{|\mathcal{A}_f|} c \cdot \mathbb{I}\left(\mathbf{a}_t^{f(i)}\right)\Big)\right], \tag{1}$$

where the expectation is over the randomness of the stochastic process and the policies, $\mathbf{a}_t^{f(i)}$ denotes the i-th feature acquisition action at timestep t, and $\mathbb{I}\left(\cdot\right)$ is an indicator function whose value equals to 1 if that feature has been acquired.

Note that the above optimization problem is very challenging: an optimal solution needs to maintain beliefs $\mathbf{b}_t$ over the state of the system at time t which is a function of partial observations obtained so far. Both the the feature acquisition policy $\pi^f(\mathbf{a}_t^f \mid \mathbf{b}_t)$ and the task policy i.e., $\pi^c(\mathbf{a}_t^c \mid \mathbf{b}_t)$ depend on this belief. The information in the belief itself can be controlled by the feature acquisition policy through querying subsets from the features $\mathbf{x}_t$ and hence the task policy and feature acquisition policy itself strongly depend on effectiveness of the feature acquisition policy.

3.2 Sequential Representation Learning with Partial Observations

We introduce a sequential representation learning approach to facilitate the task of policy training with active feature acquisition. Let $\mathbf{x}_{1:T} = (\mathbf{x}_1, ..., \mathbf{x}_T)$ and $\mathbf{a}_{1:T} = (\mathbf{a}_1, ..., \mathbf{a}_T)$ denote a sequence of observations and actions, respectively. Alternatively, we also denote these sequences as $\mathbf{x}_{\leq T}$ and $\mathbf{a}_{\leq T}$. Overall, our task of interest is to train a sequential representation learning model to learn the distribution of the full sequential observations $\mathbf{x}_{1:T}$, i.e., for both the observed part $\mathbf{x}_{1:T}^p$ and the unobserved part $\mathbf{x}_{1:T}^u$. Given only partial observations, we can perform inference only with the observed features $\mathbf{x}_{1:T}^p$. Therefore, our proposed approach extends the conventional unsupervised representation learning task to a supervised learning task, which learns to impute the unobserved features by synthesizing the acquired information and learning the model dynamics.

As such, the key underlying assumption is that learning to impute the unobserved features would result in better representations which can be leveraged by the task policy. And performing sequential representation learning, as we propose, is a more adequate choice than non-sequential modeling, for our task of interest with partial observability. Furthermore, unlike many conventional sequential representation learning models for reinforcement learning that only reason over the observation sequence $\mathbf{x}_{1:T}^p$, in our work, we take into account both the observation sequence $\mathbf{x}_{1:T}^p$ and the action sequence $\mathbf{a}_{1:T}$ for conducting inference. The intuition is that since $\mathbf{x}_{1:T}^p$ by itself consists of very limited information over the agent's underlying MDP state, incorporating the action sequence would be an informative add-on to the agent's acquired information to infer the belief state. To summarize, our proposed sequential representation model learns to encode $\mathbf{x}_{1:T}^p$ and $\mathbf{a}_{1:T}$ into meaningful latent

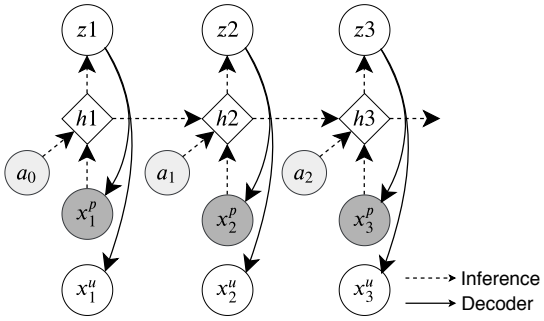


Figure 2: Observation decoder and belief inference model for the partially observable sequential VAE. Shaded nodes represent the observed variables. The inference model filters information over the partial observations and actions, to predict both the observed and unobserved features.

features, for predicting $\mathbf{x}^p_{1:T}$ and $\mathbf{x}^u_{1:T}$. The architecture of our proposed sequential representation learning model is shown in Figure 2.

Observation Decoder Let $\mathbf{z}_{1:T} = (\mathbf{z}_1, ..., \mathbf{z}_T)$ denote a sequence of latent states. We consider the following probabilistic model:

$$p_\theta(\mathbf{x}^p, \mathbf{x}^u, \mathbf{z}) = \prod_{t=1}^{T} p(\mathbf{x}^p_t, \mathbf{x}^u_t|\mathbf{z}_t)\, p(\mathbf{z}_t), \tag{2}$$

For simplicity of the notations, we assume $\mathbf{z}_0 = \mathbf{0}$. We impose a simple prior distribution over $\mathbf{z}$, i.e., a standard Gaussian prior, instead of incorporating some learned prior distribution over the latent space of $\mathbf{z}$, such as an autoregressive prior distribution like $p(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{x}^p_{1:t}, \mathbf{a}_{0:t-1})$. The reason is that using a static prior distribution results in latent representation $\mathbf{z}_t$ that is stronger regularized and more normalized than using a learned prior distribution which is stochastically changing over time. This is crucial for deriving stable policy training performance. At time t, the generation of data $\mathbf{x}^p_t$ and $\mathbf{x}^u_t$ depends on the corresponding latent variable $\mathbf{z}_t$. Given $\mathbf{z}_t$, the observed variables are conditionally independent of the unobserved ones. Therefore,

$$p(\mathbf{x}^p_t,\, \mathbf{x}^u_t|\mathbf{z}_t) = p(\mathbf{x}^p_t|\mathbf{z}_t)\, p(\mathbf{x}^u_t|\mathbf{z}_t). \tag{3}$$

Belief Inference Model During policy training we only assume access to partially observed data. This requires an inference model which takes in the past observation and action sequences to infer the latent states $\mathbf{z}$. Specifically, we present a structured inference network q_ϕ as shown in Figure 2, which has an autoregressive structure:

$$q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{a}) = \prod_t q_\phi(\mathbf{z}_t|\mathbf{x}^p_{\leq t}, \mathbf{a}_{<t}), \tag{4}$$

where $q_\phi(\cdot)$ is a function that aggregates the filtering posteriors of the history of observation and action sequences. Following the common practice in existing sequential VAE literature, we adopt a forward RNN model as the backbone for the filtering function $q_\phi(\cdot)$ (Gregor et al., 2019). Specifically, at step t, the RNN processes the encoded partial observation $\mathbf{x}^p_t$, action $\mathbf{a}_{t-1}$ and its past hidden state $\mathbf{h}_{t-1}$ to update its hidden state $\mathbf{h}_t$. Then the latent distribution $\mathbf{z}_t$ is inferred from $\mathbf{h}_t$. The belief state $\mathbf{b}_t$ is defined as the mean of the distribution $\mathbf{z}_t$. By accomplishing the supervised learning task, the belief state could provide abundant information for not only the observed sequential features, but also for the missing features, so that the policy trained over it could benefit from it and progress faster towards getting better convergent performance.

Learning We proposed to pre-train both the generative and inference models offline before learning the RL policies. In this case, we assume the access to the unobserved features, so that we can construct a supervised learning task to learn to impute unobserved features. Concretely, the pre-training task update the parameters θ, ϕ by maximizing the following variational lower-bound (Jordan et al., 1999; Kingma & Welling, 2013):

$$\begin{aligned}\log p(\mathbf{x}^p, \mathbf{x}^u) &\geq \mathbb{E}_{q_\phi}\Big[\sum_t \log p_\theta(\mathbf{x}^p_t|\mathbf{z}_t) + \log p_\theta(\mathbf{x}^u_t|\mathbf{z}_t) - \mathrm{KL}\big(q_\phi(\mathbf{z}_t|\mathbf{x}^p_{\leq t}, \mathbf{a}_{<t})\,||\,p(\mathbf{z}_t)\big)\Big] \\ &= \mathrm{ELBO}(\mathbf{x}^p, \mathbf{x}^u).\end{aligned} \tag{5}$$

By incorporating the term $\log p_\theta(\mathbf{x}_t^u|\mathbf{z}_t)$, the training of sequential VAE generalizes from an unsupervised task to a supervised task that learns the model dynamics from past observed transitions and imputes the missing features.

We perform multi-stage reinforcement learning to jointly learn the feature acquisition policy and the task policy. The VAE model is pretrained and kept fixed during policy learning. The reason for not updating VAE online is that computing the loss in Eq (5) would require the access to unobserved features and therefore, is cost intensive. The pseudocode for our proposed method is in Appendix A.

4 EXPERIMENTS

We examine the characteristics of our proposed model in the following two experimental domains: a *bouncing ball* control task with high-dimensional image pixels as input, adapted from (Fraccaro et al., 2017); a *sepsis* medical simulator fitted from real-world data (Oberst & Sontag, 2019).

Baselines For comparison, we mainly consider variants of the strong VAE baseline *beta-VAE* (Higgins et al., 2016), which works on non-time-dependent data instances. For representing the missing features, we adopt the *zero-imputing* method, proposed in (Nazabal et al., 2018) over the unobserved features. Thus, we denote the VAE baseline as *NonSeq-ZI*. We train the VAE with either the *full* loss over the entire features, or the *partial* loss which only applies to the observed features (Ma et al., 2019). We denoted our proposed sequential VAE model for POMDPs as *Seq-PO-VAE*. All the VAE-based approaches adopt an identical policy architecture. Detailed information on the model architecture is presented in appendix.

Data Collection To pre-train the VAE models, data generated by a non-random policy is unavoidably needed to incorporate abundant dynamics information. For both tasks, we collect a small scale dataset of 2000 trajectories, where half of the data is collected from a random policy and the the other half from a policy which better captures the state space that would be encountered by a learned model (e.g., by training a data collection policy end-to-end or using human generated trajectories). The simple mixture of dataset works very well on both tasks without the need of further fine-tuning the VAEs. We also create a testing set that consists of 2000 trajectories to evaluate the models.

4.1 BOUNCING BALL$^+$

Task Settings We adapted the original *bouncing ball* experiment presented in (Fraccaro et al., 2017) by adding a navigation objective and introducing control actions. Specifically, a ball moves in a 2D box and at each step, a binary image of size 32×32 showing the box and the ball. Initially, the ball appears at a random position in the upper left quadrant, and has a random velocity. The objective is to control the ball to reach a fixed target location set at $(5, 25)$. We incorporate five RL actions: a null action and four actions for changing the velocity of the ball in either the x or y direction with a fixed scale: $\{\Delta V_x : \pm 0.5,\ \Delta V_y : \pm 0.5,\ null\}$. A reward of 1.0 is issued if the ball reaches its target location. Each episode runs up to 50 time steps.

Table 1: Imputing loss evaluated on missing features for *Bouncing Ball*$^+$ and *Sepsis*. Each setting is evaluated for 3 random seeds.

VAE MODEL	BOUNCING BALL$^+$ (NLL)	SEPSIS (MSE)
NONSEQ-ZI (PARTIAL)	0.5846 ($\pm$ 0.0544)	1.1954 ($\pm$0.2521)
NONSEQ-ZI (FULL)	0.0728 ($\pm$ 0.0004)	0.5217 ($\pm$ 0.0016)
SEQ-PO-VAE (OURS)	**0.0284** ($\pm$ 0.0002)	**0.1724** ($\pm$0.0064)

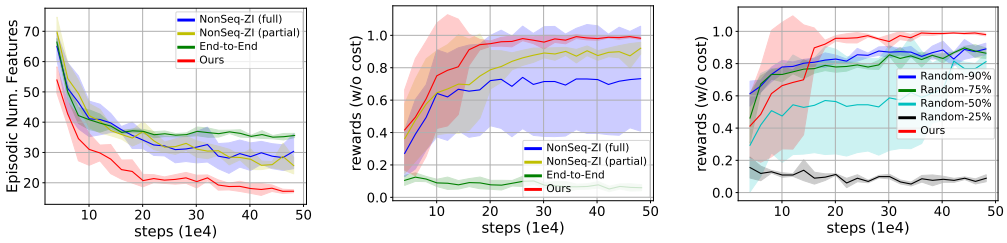


Figure 3: Performance curves on the *bouncing ball*$^+$ domain: **a**: episodic number of observations acquired by the π^f; **b**: task rewards w/o cost. Our proposed method outperforms the non-sequential baselines in learning the task as well as acquiring less observations; **c**: Ablation study on *bouncing ball*$^+$ to illustrate the effect of learning the feature acquisition policy. Each method is run with 10 random seeds. Our proposed approach outperforms the random baseline significantly in terms of task performance.

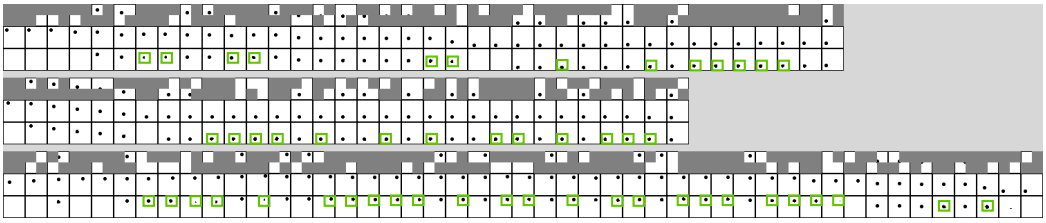


Figure 4: *Seq-PO-VAE* reconstruction for the online trajectories upon convergence (better to view enlarged). Each block of three rows corresponds to the results for one trajectory. In each block, the three rows (top-down) correspond to: (1) the partially observable input selected by acquisition policy; (2) the ground-truth full observation; (3) reconstruction from *Seq-PO-VAE*. The green boxes remark the frames where ball is not observed but our model could impute its location. Key takeaways: (1) our learned acquisition policy captures model dynamics ; (2) *Seq-PO-VAE* effectively impute the missing features (i.e., ball can be reconstructed even when they are unobserved from consequent frames).

Representation Learning Results We evaluate the missing feature imputing performance of each VAE model in terms of negative log likelihood (nll) loss and present results in Table 1. We notice that our proposed model yields to significantly better imputing result than all the other baselines. This reveals the fact that our proposed sequential VAE model can efficiently capture the environment dynamics and learn meaningful information over the missing features. Such effect is vital in determining the policy training performance in AFA-POMDP, since the policy is conditioned on the VAE latent features. We also demonstrate sample trajectories reconstructed by different VAE models in the Appendix. The result shows that our model learns to impute significant amount of missing information given the partially observed sequence.

Policy Training Results We evaluate the policy training performance in terms of episodic number of acquired observations and the task rewards (w/o cost). The results are presented in Figure 3 (a) and (b), respectively. First, we notice that the *end-to-end* method is vital and fails to learn task skills under the given feature acquisition cost. However, the VAE-based representation learning methods manage to learn the navigation skill under the same cost setting. This verifies our assumption that representation learning could bring significant benefit to the policy training under the AFA-POMDP scenario. Furthermore, we also notice that the joint policies trained by *Seq-PO-VAE* can develop the target navigation skill at a much faster pace than the non-sequential baselines. Our method also converges to a standard where much less feature acquisition is required to perform the task.

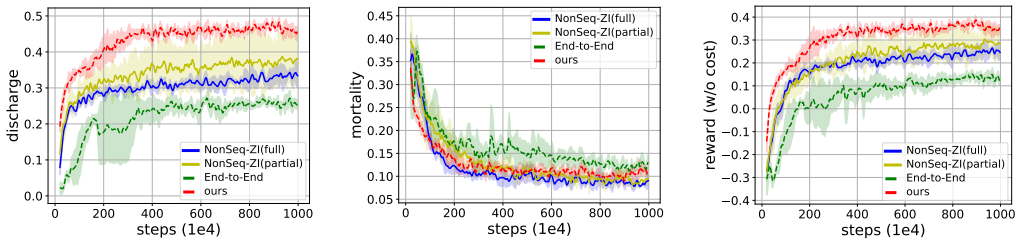


Figure 5: Performance curves for the compared approaches on *Sepsis*. The curves are derived under cost value of 0.01. Overall, our method converges to treatment policy with substantially better reward compared to the baselines.

We also show that our proposed method can learn meaningful feature acquisition policies. To this end, we show three sampled trajectories upon convergence of training in Figure 4. From the examples, we notice that our feature acquisition policy acquires meaningful features with a majority grasping the exact ball location. Thus, it demonstrates that the feature acquisition policy adapts to the dynamics of the problem and learns to acquire meaningful features. We also show the actively learned feature acquisition policy works better than random acquisition. From the results in Figure 4 (c), our method converges to better standard than random policies with considerably high selection probabilities.

4.2 Sepsis Medical Simulator

Task Setting Our second evaluation domain adopts a medical simulator for treating sepsis among ICU patients, proposed in (Oberst & Sontag, 2019). Overall, the task is to learn to apply three *treatment* actions to the patient, i.e, $\{$*antibiotic*, *ventilation*, *vasopressors*$\}$. The state space consists of 8 features: 3 of them indicate the current *treatment* state for the patient; 4 of them are the *measurement* states over *heart rate*, *sysBP rate*, *percoxyg state* and *glucose level*; the rest is an index specifying the patent's *diabetes* condition. The feature acquisition policy learns to actively select the *measurement* features. Each episode runs for up to 30 steps. The patient will be discharged if his/her *measurement* states all return to normal values. An episode terminates upon mortality or discharge, with a reward -1.0 or 1.0.

Representation Learning Result We evaluate the imputation performance for each VAE model on the testing dataset. The loss is evaluated in terms of MSE, presented in Table 1. Our proposed method leads to the lowest MSE loss compared to the baselines. The result reveals that our proposed sequential VAE could promisingly learn model dynamics for tasks with stochastic transitions.

Policy Training Result We show the policy training results for *Sepsis* in Figure 5. Overall, our proposed method results in substantially better task reward compared to all baselines. Noticeably, the learning of discharge for our method progresses significantly faster than baseline approaches and converges to substantially better values. The result shows that our method can be trained in a much more sample efficient way. Moreover, upon convergence, our model outperforms the best non-sequential VAE baseline with a gap of $> 5\%$ for discharge ratio. For all the evaluation metrics, we notice that VAE-based representation learning models outperform the end-to-end baseline by significant margins. This indicates that efficient representation learning may be crucial for deriving satisfying task performance in AFA-POMDP setting. The result also reveals that learning to impute missing features contributes greatly to improve the policy training performance for AFA-POMDP.

Ablation: Efficacy of Active Feature Acquisition We study the effect of actively learning sequential feature acquisition strategy with RL. To this end, we compare our method with a baseline that randomly acquires features. We evaluate our method under different cost values, and the results are shown in Figure 6. From the results, we notice that there is a clear cost-performance trade-off, i.e., a

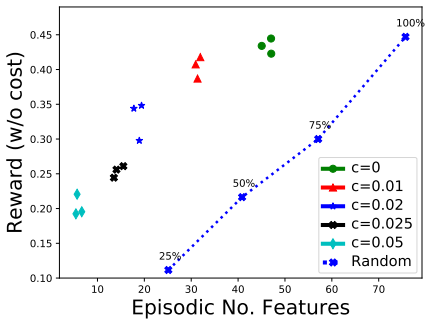


Figure 6: The task reward and episodic no. feature acquisitions derived upon convergence for our method, when training our model under different cost values. We also present the results for a baseline that trains the task policy under a random feature acquisition strategy. The actively learned feature acquisition leads to significant advantage compared to random acquisition.

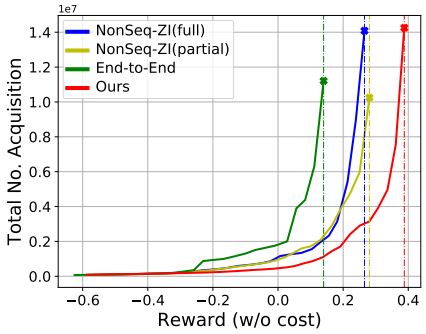


Figure 7: This figure shows the total number of features acquired (y-value) to obtain certain reward standard (x-value), when training the task policy using different representation learning approaches. The presented methods adopt a cost $c = 0.01$. Our method results in the best achievable reward (max x-value) with lowest slope.

higher feature acquisition cost results in feature acquisition policies that obtain fewer observations, with a sacrifice of task performance. Overall, our acquisition method results in significantly better task performance than the random acquisition baselines. Noticeably, with the learned active feature acquisition strategy, we acquire only about half of the total number of features (refer to the value derived by *Random*-100%) to obtain comparable task performance. Also, we notice that the specified cost has a very clear impact on the final task performance, i.e., the number of acquired features per episode decreases significantly as the cost increases. Thereby, our proposed solution can promisingly compute feature acquisition policies that meet different budgets.

Ablation: Impact on Total Acquisition Cost For different representation learning methods, we also investigate the total number of features acquired at different stage of training. The results are shown in Figure 7. As expected, to obtain better task policies, the models need to take longer training steps and thus the total feature acquisition cost would increases accordingly. We notice that policies trained by our method result in the highest convergent task performance (max x-value). Given a certain performance level (same x-value), our method consumes substantially less total feature acquisition cost (y-value) than the others. We also notice that the overall feature acquisition cost increases with a near exponential trend. Therefore, it is essential to train the policy for AFA-POMDP with advanced representation learning method, so that the feature acquisition cost could be reduced.

5 Conclusion

We present a novel AFA-POMDP framework that jointly learns the task policy and the active feature acquisition strategy with a unified reinforcement learning formalism. We introduce a novel sequential VAE model to facilitate policy training under partial observability. We demonstrate that imputing missing features via learning model dynamics could significantly benefit policy training with partial observability. Our proposed model, by efficiently synthesizing the sequential information and imputing missing features, can significantly outperform conventional representation learning baselines and leads to policy training with significantly better sample efficiency and obtained solutions. Future work may investigate whether our proposed model could be applied to more diverse and complex application domains. Another promising direction is to integrate our framework with model-based planning for further reducing the feature acquisition cost.

REFERENCES

Anthony R Cassandra. A survey of pomdp applications. In *Working notes of AAAI 1998 fall symposium on planning with partially observable Markov decision processes*, volume 1724, 1998.

Abhishek Das, Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. Learning cooperative visual dialog agents with deep reinforcement learning. In *Proceedings of the IEEE international conference on computer vision*, pp. 2951–2960, 2017.

Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pp. 49–58, 2016.

Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Advances in Neural Information Processing Systems*, pp. 3601–3610, 2017.

Wenbo Gong, Sebastian Tschiatschek, Sebastian Nowozin, Richard E. Turner, José Miguel Hernández-Lobato, and Cheng Zhang. Icebreaker: Element-wise efficient information acquisition with a bayesian deep latent gaussian model. In *Annual Conference on Neural Information Processing Systems*, pp. 14791–14802, 2019.

Karol Gregor, George Papamakarios, Frederic Besse, Lars Buesing, and Theophane Weber. Temporal difference variational auto-encoder. *ICLR*, 2019.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2016.

Irina Higgins, Arka Pal, Andrei Rusu, Loic Matthey, Christopher Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. Darla: Improving zero-shot transfer in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1480–1490, 2017.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *ICLR*, 2016.

Yuan Ling, Sadid A Hasan, Vivek Datla, Ashequl Qadir, Kathy Lee, Joey Liu, and Oladimeji Farri. Learning to diagnose: assimilating clinical narratives using deep reinforcement learning. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 895–905, 2017.

Chao Ma, Sebastian Tschiatschek, Konstantina Palla, José Miguel Hernández-Lobato, Sebastian Nowozin, and Cheng Zhang. EDDI: efficient dynamic discovery of high-value information with partial VAE. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 4234–4243, 2019.

Stefan Mathe, Aleksis Pirinen, and Cristian Sminchisescu. Reinforcement learning for visual object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2894–2902, 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Alfredo Nazabal, Pablo M Olmos, Zoubin Ghahramani, and Isabel Valera. Handling incomplete heterogeneous data using vaes. *arXiv preprint arXiv:1807.03653*, 2018.

Michael Oberst and David Sontag. Counterfactual off-policy evaluation with gumbel-max structural causal models. In *ICML*, 2019.

Yu-Shao Peng, Kai-Fu Tang, Hsuan-Tien Lin, and Edward Chang. Refuel: Exploring sparse features in deep reinforcement learning for fast disease diagnosis. In *Advances in Neural Information Processing Systems*, pp. 7322–7331, 2018.

Hajin Shim, Sung Ju Hwang, and Eunho Yang. Joint active feature acquisition and classification with variable-size set encoding. In *Advances in Neural Information Processing Systems*, pp. 1368–1378, 2018.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

Giulia Vezzani, Abhishek Gupta, Lorenzo Natale, and Pieter Abbeel. Learning latent state representation for speeding up exploration. *arXiv preprint arXiv:1905.12621*, 2019.

Jinsung Yoon, James Jordon, and Mihaela van der Schaar. Invase: Instance-wise variable selection using neural networks. In *ICLR*, 2018.

Sara Zannone, José Miguel Hernández-Lobato, Cheng Zhang, and Konstantina Palla. Odin: Optimal discovery of high-value information using model-based deep reinforcement learning. In *ICML Real-world Sequential Decision Making Workshop*, 2019.

Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew J Johnson, and Sergey Levine. Solar: Deep structured latent representations for model-based reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

APPENDIX

This appendix is organized as follows:

- **Sec A**: the detailed algorithm.
- **Sec B**: experimental settings and additional results on the *Bouncing Ball* domain.
- **Sec C**: experimental settings and additional results on the *Sepsis* domain.

A RL WITH ACTIVE FEATURE ACQUISITION ALGORITHM

Algorithm 1 RL with Active Feature Acquisition

Input: learning rate $\alpha > 0$, dataset $\mathcal{D}$
Initialize RL policy π_f, π_c, VAE parameters θ, ϕ.
Train VAE on dataset $\mathcal{D}$ using Eq (5).
while Not Converge **do**
  Reset the environment.
  Initialize null observation $\mathbf{x}_1^p = \emptyset$, feature acquisition action $\mathbf{a}_0^f$ and control action $\mathbf{a}_0^c$.
  for $i = 1$ **to** T **do**
    Compute representation with VAE: $\mathbf{b}_t = q_\phi(\mathbf{x}^p_{\leq t}, \mathbf{a}_{<t})$.
    Sample a feature acquisition action $\mathbf{a}_t^f \sim \pi_f(\mathbf{b}_t)$ and a control action $\mathbf{a}_t^c \sim \pi_c(\mathbf{b}_t)$.
    Step the environment and receive partial features, reward and terminal: $\mathbf{x}^p_{t+1}, r_t, \text{term} \sim \text{env}(\mathbf{a}_t^f, \mathbf{a}_t^c)$
    Compute cost $c_t = \sum_i c \cdot \mathbb{I}(\mathbf{a}_t^{f(i)})$.
    Save the transitions $\{\mathbf{b}_t, \mathbf{a}_t^f, \mathbf{a}_t^c, r_t, c_t, \text{term}\}$.
    if term **then**
      break
    end if
  end for
  Update π_f, π_c using the saved transitions with an RL algorithm under learning rate α.
end while

B BOUNCING BALL$^+$

B.1 TASK SPECIFICATION

The task consists of a ball moving in a 2D box of size 32×32 pixels. The radius of the ball equals to 2 pixels. At each step, a binary image is returned as an observation of the MDP state. At the beginning of every episode, the ball starts at a random position in the *upper left* quadrant (sampled uniformly). The initial velocity of the ball is randomly defined as follows: $\vec{v} = [V_x, V_y] = 4 \cdot \tilde{\vec{v}}/\|\tilde{\vec{v}}\|$, where the x- and y-component of $\tilde{\vec{v}}$ are sampled uniformly from the interval $[-0.5, 0.5]$. There is a navigation target set at $(5, 25)$ pixels, which is in the *lower left* quadrant. The navigation is considered to be successful if the ball reaches the specified target location within a threshold of 1 pixel along both x/y-axis.

The action spaces is defined as follows. There are five task actions $\mathcal{A}^c$:

- Increase velocity leftwards, i.e., change V_x by -0.5
- Increase velocity rightwards, i.e., change V_x by $+0.5$
- Increase velocity downwards, i.e., change V_y by $+0.5$
- Increase velocity upwards, i.e., change V_y by -0.5
- Keep velocities unchanged

The maximum velocity along the x/y-axis is 5.0. The velocity will stay unchanged if it exceeds this threshold. The feature acquisition action $\mathbf{a}^f \in \mathcal{A}^f$ is specified as acquiring the observation of a subset

of the quadrants (this also includes acquiring the observation of all 4 quadrants). Thus, the agent can acquire $0-4$ quadrants to observe. Each episode runs up to 50 steps. The episode terminates if agent reaches the target location.

B.2 Implementation Details

For all the compared methods, *Zero-Imputing* (Nazabal et al., 2018) is adopted to fill in missing features with a fixed value of 0.5.

End-to-End The end-to-end model first processes the imputed image by 2 *convolutional* layers with filter sizes of 16 and 32, respectively. Each *convolutional* layer is followed by a *ReLU* activation function. Then the output is passed to a *fully connected* layer of size 1024. The weights for the *fully connected* layer are initialized by *orthogonal weights initialization* and the biases are initialized as zeros.

NonSeq-ZI The non-sequential VAE models first process the imputed image by 2 *convolutional* layers with filter sizes of 32 and 64, respectively. Each *convolutional* layer is followed by a *ReLU* activation function. Then the output passes through a *fully connected* layer of size 256, followed by two additional *fully connected* layers of size 32 to generate the mean and variance of a Gaussian distribution. To decode an image, the sampled code first passes through a *fully connected* layer with size 256, followed by 3 *convolutional* layers with filters of 32, 32, and nc and strides of 2, 2 and 1, respectively, where nc is the *channel* size that equals to 2 for the binary image. There are two variants for *NonSeq-ZI*: one employs the *partial* loss that is only for the observed variables; the other employs the *full* loss that is computed on all the variables, i.e., the ground-truth image with full observation is employed as the target to train the model to impute the missing features. The hyperparameters for training *NonSeq-ZI* are summarized in Table 2.

	Hyperparameter				
	β (KL weight)	KL reduction	Loss reduction	learning rate	epochs
NonSeq-ZI (partial)	1.0	sum	sum	1e-4	1k
NonSeq-ZI (full)	1.0	sum	sum	1e-4	1k
Seq-PO-VAE (ours)	1.0	sum	sum	5e-4	2k

Table 2: Hyperparameter settings for training VAE models on the *Bouncing Ball*$^+$ dataset.

Seq-PO-VAE (ours) At each step, the *Seq-PO-VAE* takes an imputed image and an action vector of size 9 as input. The imputed image is processed by 3 *convolutional* layers with filter size 32 and stride 2. Each *convolutional* layer employs *ReLU* as its activation function. Then the output passes through a *fully connected* layer of size 32 to generate a latent representation for the image $\mathbf{f}_x$. The action vector passes through a *fully connected* layer of 32 to generate latent representation for the action $\mathbf{f}_a$. Then the image and action features are concatenated and augmented to form a feature vector $\mathbf{f}_c = [\mathbf{f}_x, \mathbf{f}_a, \mathbf{f}_x * \mathbf{f}_a]$, where $[\cdot]$ denotes *concatenation* of features. Then $\mathbf{f}_c$ is fed to *fully connected* projection layers of size 64 and 32, respectively. The output is then fed to an *LSTM* module, with latent size of 32. The output $\mathbf{h}_t$ of *LSTM* is passed to two independent *fully connected* layers of size 32 for each to generate the mean and variance for the Gaussian distribution filtered from the sequential inputs. To decode an image, the model adopts *deconvolutional* layers that are identical to those for *NonSeq-ZI*. The hyperparameters for training *Seq-PO-VAE* are shown in Table 2.

LSTM-A3C We adopt LSTM-A3C (Mnih et al., 2016) to train the RL policy. The policy takes the features derived from the representation learning module as input. For the VAE-based methods, the input features are passed through a *fully connected* layer of size 1024. Then the features are fed to an *LSTM* with 1024 units. The output of the *LSTM* is fed to three independent *fully connected* layers to generate the estimations for value, task policy and feature acquisition policy. We adopt *normalized column* initialization for all the *fully connected* layers and the biases for the *LSTM* module are set to be zero.

B.3 DATA COLLECTION

To train the VAEs, we prepare a training set that consists of 2000 trajectories. Half of the trajectories are derived from a random policy and the other half is derived from a policy learned from end-to-end method. To train the end-to-end method, we employ a cost of 0.01 over first 2m steps and then increase it to 0.02 for the following 0.5m steps. All the VAE models are evaluated on a test dataset that has identical size and data distribution as the training dataset. We present the best achieved task performance of the data collection policy (*End-to-End*) and our representation learning approach in Table 5. We notice that our proposed method, by employing an advanced representation model, leads to significantly better feature acquisition policy than *End-to-End* (smaller number of observations while achieving similar or better reward).

	Model	
	End-to-End	Ours
Average # of observations per episode	17.94	**8.24**
Task reward	1.0	1.0

Table 3: Task performance for the data collection policy and our method on *Bouncing Ball*$^{+}$.

B.4 IMPUTING MISSING FEATURES VIA LEARNING MODEL DYNAMICS

We present an illustrative example to demonstrate the process of imputing missing features and the role of learning model dynamics. To this end, we collect trajectories under an *End-to-End* policy (the choice of the underlying RL policy is not that important since we just want to derive some trajectory samples for the VAE models to reconstruct) and use different VAE models to impute the observations.

From the results presented in Figure 9, we observe that under the partially observable setting with missing features, the latent representation derived from our proposed method provides abundant information as compared to only using information from a single time step and thereby offers significant benefit for the policy model to learn to acquire meaningful features/gain task reward.

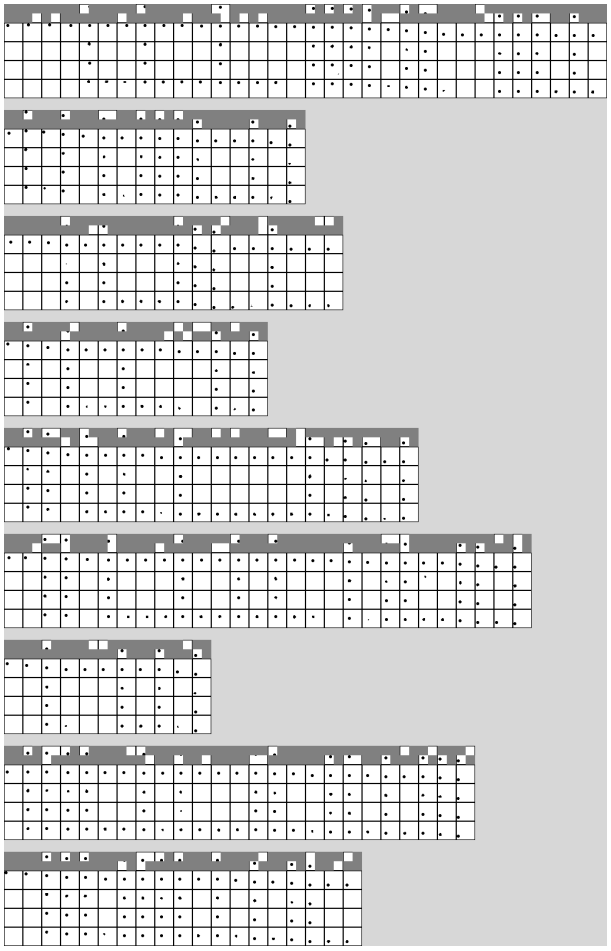
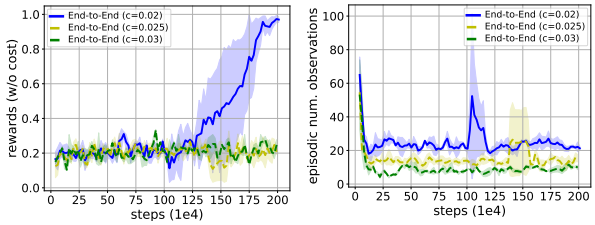


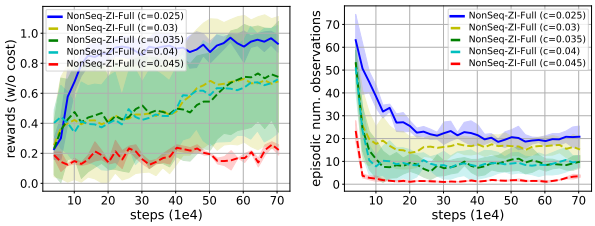
Figure 8: Imputation results for different VAE models. We select 9 trajectories obtained from the trained *End-to-End* policy. Each block corresponds to the results for one trajectory (better to view enlarged). The five rows in one block are (top-down): (1) partial observations acquired by the agent; (2) ground-truth image with full observation; (3) Imputation by *NoSeq-ZI (partial)*; (4) Imputation by *NoSeq-ZI (full)*; (5) Imputation by *Seq-PO-VAE (ours)*. Our model can often successfully predict the balls location even if it is not present in the acquired observation. Hence it successfully employs its learned knowledge of the dynamics. In contrast, the non-sequential model (obviously) fails to predict the balls location when the ball is not present in the observation.

B.5 INVESTIGATION ON COST-PERFORMANCE TRADE-OFF

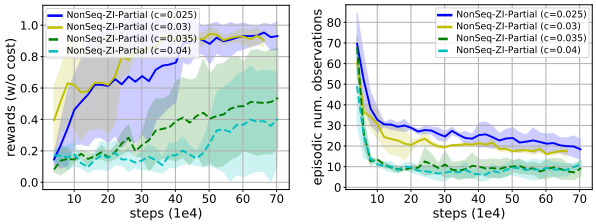
We perform a case study on investigating the cost-performance trade-off for each representation learning method, presented in Figure 9. Apparently, as we increase the cost, the exploration-exploitation task becomes more challenging and each compared method has its own upper bound on the cost above which it fails to learn an effective task policy while acquiring minimum observation. First, we notice that the *End-to-End* model takes a long time to progress in learning task skills, while the VAE-based models can progress much faster. Among the VAE-based methods, we notice that our proposed method (Figure 9(d)) can achieve as low as 8 observations whereas the baselines *NonSeq-ZI (Full)* (Figure 9(b)) and *NonSeq-ZI (partial)* (Figure 9(c)) achieve a standard of $\sim$20 (lowest point among the *solid* lines). Thus, we could conclude that our proposed approach can significantly benefit the cost-sensitive policy training and lead to a policy which acquires much fewer observations while still succeeding in terms of task performance.



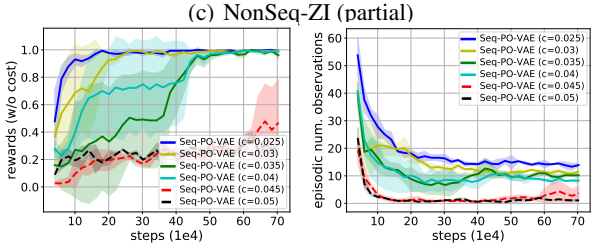
(a) End-to-End



(b) NonSeq-ZI (full)



(c) NonSeq-ZI (partial)



(d) Seq-PO-VAE (ours)

Figure 9: Cost-performance trade-off investigation. Each row corresponds to the performance in terms of task reward and episodic number of acquisitions obtained for a specific method (see legends). Each curve is derived from 3 independent runs. We use dotted lines to indicate those instances when the task learning does not always succeed. Thus, the best achievable number of observations should be referred to as the lowest curve among the *solid* lines.

C SEPSIS MEDICAL SIMULATOR

C.1 TASK SPECIFICATIONS

For this task we employ a Sepsis simulator proposed in previous work (Oberst & Sontag, 2019). The task is to learn to apply three *treatment* actions for Sepsis patients in intensive care units, i.e., $\mathcal{A}^c = \{$*antibiotic*, *ventilation*, *vasopressors*$\}$. At each time step, the agent selects a subset of the *treatment* actions to apply. The state space consists of 8 features: 3 of them specify the current *treatment* status; 4 of them specify the *measurement* status in terms of *heart rate*, *sysBP rate*, *percoxyg stage* and *glucose level*; the remaining one is a categorical feature indicating the patent's antibiotic status. The feature acquisition actively selects a subset among the *measurement* features for observation, i.e., $\mathcal{A}^f = \{$*heart rate*, *sysBP rate*, *percoxyg state*, *glucose level*$\}$. The objective for learning a active feature acquisition strategy is to help the decision making system to reduce *measurement* cost at a significant scale.

C.2 IMPLEMENTATION DETAILS

For all the compared methods, we adopt *Zero-Imputing* (Nazabal et al., 2018) to fill in missing features. In particular, a fixed value of -10 which is outside the range of feature values is used to impute missing values.

End-to-End The end-to-end model first processes the imputed state by 3 *fully connected* layers of size 32, 64 and 32, respectively. Each *fully connected* layer is followed by a *ReLU* activation.

NonSeq-ZI The VAE model first processes the imputed state by 2 *fully connected* layers with size 32 and 64, with the first *fully connected* layer being followed by *ReLU* activation functions. Then the output is fed into two independent *fully connected* layers of size 10 for each, to generate the mean and variance for the Gaussian distribution. To decode the state, the latent code is first processed by a *fully connected* layer of size 64, then fed into three *fully connected* layers of size 64, 32, and 8. The intermediate *fully connected* layers employ *ReLU* activation functions. Also, we adopt two variants for *NonSeq-ZI*, trained under either *full* loss or *partial* loss. The details of the hyperparameter settings used for training are presented in Table 4.

	Hyperparameter				
	β (KL weight)	KL reduction	Loss reduction	learning rate	epochs
NonSeq-ZI (partial)	0.01	sum	sum	1e-4	1k
NonSeq-ZI (full)	0.01	sum	sum	1e-4	1k
Seq-PO-VAE (ours)	0.01	sum	sum	1e-3	1k

Table 4: Hyperparameter settings for training VAE models on the *Sepsis* dataset.

Seq-PO-VAE (ours) At each time step, the inputs for state and action are first processed by their corresponding projection layers. The projection layers for the state consists of 3 *fully connected* layers of size 32, 16 and 10, where the intermediate *fully connected* layers are followed by a *ReLU* activation function. The projection layer for the action input is a *fully connected* layer of size 10. Then the projected state feature $\mathbf{f}_c$ and action feature $\mathbf{f}_a$ are combined in the following manner: $\mathbf{f}_c = [\mathbf{f}_x, \mathbf{f}_a, \mathbf{f}_x * \mathbf{f}_a]$. $\mathbf{f}_c$ is passed to 2 *fully connected* layers of size 64 and 32 to form the input to the *LSTM* module. The output $\mathbf{h}_t$ of the *LSTM* is fed to two independent *fully connected* layers of size 10 to generate the mean and variance for the Gaussian distribution. The decoder for *Seq-PO-VAE* has identical architecture as *NonSeq-ZI*. The details for training *Seq-PO-VAE* are presented in Table 4.

LSTM-A3C The LSTM-A3C (Mnih et al., 2016) takes encoded state features derived from the corresponding representation model as its input. The encoded featuresare fed into an *LSTM* with size 256. Then the $\mathbf{h}_t$ for the *LSTM* is fed to three independent *fully connected* layers, to predict the state value, feature acquisition policy and task policy. *Normalized column* initialization is applied to all *fully connected* layers. The biases for the *LSTM* and *fully connected* layers are initialized as zero.

C.3 DATA COLLECTION

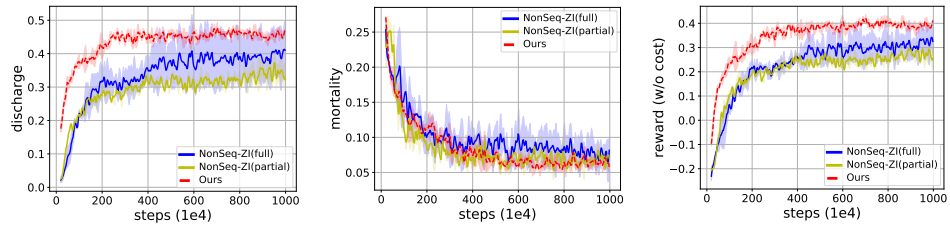
To train the VAEs, we prepare a training set that consists of 2000 trajectories. Half of the trajectories are derived from a random policy and the other half is derived from a policy learned from the *End-to-End* method with cost 0.0. All the VAE models are evaluated on a test dataset that consists of identical size and data distribution as the the training dataset. We present the task treatment reward obtained by our data collection policy derived from the *End-to-End* method and that obtained by our proposed method in Table 5. Noticeably, by performing representation learning, we obtained much better treatment reward as compared to the data collection policy, which demonstrates the necessity of performing representation learning.

	Model	
	End-to-End	Ours
Treatment Reward	0.35	**0.45**

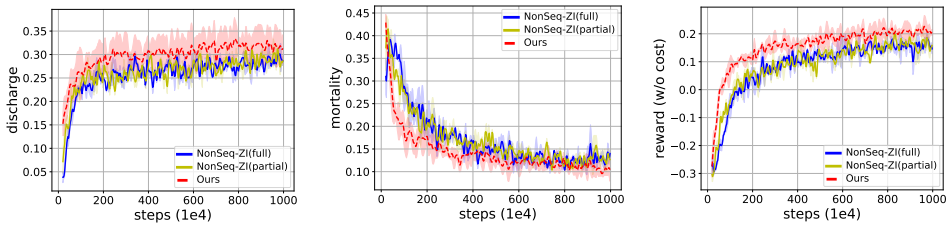
Table 5: Task performance for the data collection policy and our proposed method on *Sepsis*.

C.4 MORE COMPARISON RESULT UNDER DIFFERENT VALUES FOR COST

We present additional experiment results that compare our proposed method and the non-sequential baselines under the cost values $\{0, 0.025\}$. The results for cost value of 0.01 are shown in the main paper. Overall, under all the cost settings, our method leads to significantly better discharge ratio and task reward compared to the baselines.



(a) Cost=0



(b) Cost=0.025

Figure 10: Comparison result between our proposed method and the non-sequential VAE baseline models under different values for cost. Each curve is derived from 3 independent runs.

Also, we demonstrate the cost-performance trade-off on *Sepsis* domain. By increasing the value of cost, we could obtain feature acquisition policy that acquires substantially decreased amount of features within each episode.

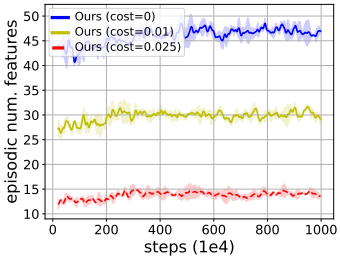


Figure 11: Average number of observations acquired in each episode when training our proposed model under different cost values.

C.5 Illustrative Examples for Missing Feature Imputation in *Sepsis*

We present two illustrative examples in Figure 12 to demonstrate how imputing missing features via learning model dynamics would help the decision making with partial observability in *Sepsis* domain. The policy training process with partial observability could only access very limited information, due to the employment of active feature acquisition. Under such circumstances, imputing the missing features would offer much more abundant information to the decision making process. From the results shown in Figure 12, our model demonstrates considerable accuracy in imputing the missing features, even though it is extremely challenging to perform the missing feature imputation task given the distribution shift from the data collection policy and the online policy. The imputed missing information would be greatly beneficial for training the task policy and feature acquisition policy.

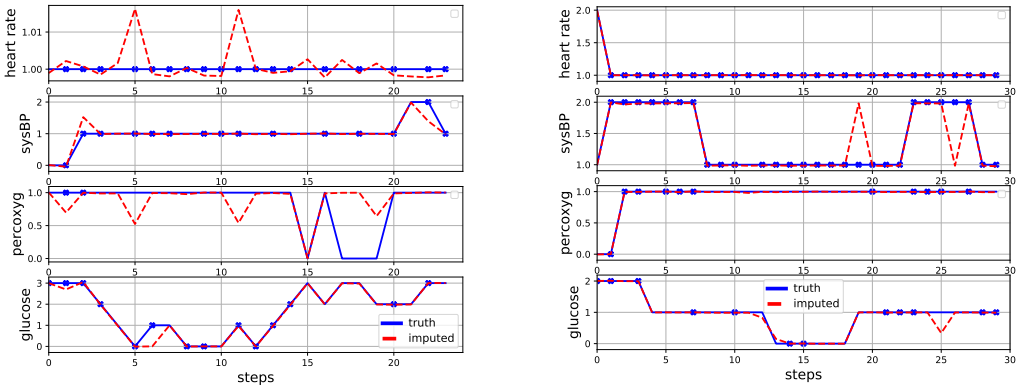


Figure 12: Two example trajectories for illustrating how our method works on the *Sepsis* medical domain. The acquisition policy is trained with a cost of 0. Each block corresponds to one trajectory and the four rows correspond to the four *measurement* features being considered for active feature acquisition. Each dot indicates the employment of feature acquisition on the corresponding *measurement* feature at the presented time point. In each trajectory, we demonstrate the ground-truth signal over time as well as the imputed signal over time predicted by our proposed *Seq-PO-VAE* model. By imputing the missing features via learning model dynamics, our proposed method could offer much more informative representation for the policy training compared to the non-sequential VAE baselines, and thus significantly benefit the policy training with partial observability.