Reinforcement Learning with Efficient Active Feature Acquisition

Anonymous Author(s)
Affiliation
Address
email

Abstract

Solving real-life sequential decision making problems under partial observability involves an exploration-exploitation problem. An agent needs to gather information about the state of the world for making rewarding decisions. However, in real-life, acquiring 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 while reducing the cost for information acquisition. In this paper, we propose a model-based reinforcement learning framework that learns an active feature acquisition policy to solve the exploration-exploitation problem during its execution. Key to the success is a novel sequential variational auto-encoder. We demonstrate the efficacy of our proposed framework in a control domain as well as using a medical simulator, outperforming natural baselines and resulting 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 [2, 16], robotics control [3, 34], medical diagnosis [13, 22] and computer games [19, 25]. These models are typically trained on large amounts of data with a fixed set of available features, and when these models are deployed, they are assumed to operate on data with the same features. However, in many real-world applications, the fundamental assumption that the same features are always readily available during deployment does not hold. For instance, consider a medical support system for monitoring and treating patients during their stay at hospital. To provide the best possible treatment, the system might need to perform several measurements of the patient over time. However, some of these measurements could be costly or pose a health risk. That is, at the deployment, the system should function with minimal and carefully selected features while during training more features might have been available.

In this paper, we consider the challenging problem of learning effective sequential decision making policies when the cost of feature acquisition cannot be neglected. To be successful, we need to learn policies which acquire the information required for making the task related decisions in the most cost efficient way. For simplicity, we can think of the policies 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. As a consequence, these two policies are typically intimately connected, i.e., the acquisition policy must collect features such that the task policy can take good actions, and the task policy needs to enable the acquisition policy to collect informative features by transiting to appropriate states. As such, our work tackles a partially observable policy learning problem with the following two distinguishing properties compared to the most commonly studied problems. First, 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 controlled by the acquisition policy. Thus, the resulting POMDP is different from typically considered POMDPs in

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

RL literature [1] 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. Second, the learning of the acquisition policy introduces an additional dimension to the exploration-exploitation problem: each execution of the acquisition and task policy needs to solve an exploration-exploitation problem.

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 exploited the use of reinforcement learning for sequential feature acquisition tasks [24, 32], 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.

We approach this problem and present a framework which tackles the problem from a representation learning perspective. In particular, we make the following contributions: 1. We propose a general solution for learning reinforcement learning policies with active feature acquisition. Our proposed approach simultaneously learns reinforcement learning policies for reward optimization and active feature acquisition, approximately solving a challenging combinatorial problem. 2. We present a novel sequential representation learning approach to account for the encoding of the partially observed states based on sequential variational autoencoders (VAE). 3. We present experiment results on an image-based control task as well as a medical simulator fitted from real-life data, where our method shows clear improvements over natural baselines.

2 Methodology

2.1 Problem Setting

In this section, we formalize our problem setting. To this end, we define the *active feature acquisition POMDP* (AFA-POMDP), a rich class of discrete-time stochastic control processes.

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* $\mathcal{A} = \mathcal{A}^c \times \mathcal{A}^f$ *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* $\mathbf{a} = (\mathbf{a}^c, \mathbf{a}^f)$ *in state* $s \in \mathcal{S}$ *to a distribution* $P_{\mathcal{S}}$ *over next states. In each state* s*, the agent observes the features* $\mathbf{x}^p$ *which are a subset of the features* $\mathbf{x} = (\mathbf{x}^p, \mathbf{x}^u) \sim \mathcal{O}(s)$ *selected by the agent taking feature acquisition action* $\mathbf{a}^f$*, 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* $\mathcal{R} \colon \mathcal{S} \times \mathcal{A}^c \to \mathbb{R}$ *and pays a cost of* $\mathcal{C} \colon \mathcal{S} \times \mathcal{A}^f \to \mathbb{R}_{\geq 0}$ *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 $\mathbf{a}^f \in \mathcal{A}^f$. Furthermore, we assume in the following that transitions depend 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 $\mathbf{a}^{f'}, \mathbf{a}^f \in \mathcal{A}^f$. This assumption can be a reasonable approximation for instance for medical settings in which tests are non-invasive. 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 easily adapted to feature-dependent costs.

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:

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

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

Remarks Any AFA-POMDP corresponds to a POMDP in which the reward is defined appropriately from $\mathcal{R}$ and $\mathcal{C}$ and observations depend on the taken joint action. Through enabling to query subsets of observations, the feature acquisition action space $\mathcal{A}^f$ is exponential in the number of features.

2.2 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 $\mathbf{x}_{1:T} = \mathbf{x}_{\leq T} = (\mathbf{x}_1, ..., \mathbf{x}_T)$ and $\mathbf{a}_{1:T} = \mathbf{a}_{\leq T} = (\mathbf{a}_1, ..., \mathbf{a}_T)$ denote a sequence of observations and actions, respectively. We aim to train a sequential representation learning model for the full sequential observations $\mathbf{x}_{1:T}$, i.e., for both the observed part $\mathbf{x}^p_{1:T}$ and the unobserved part $\mathbf{x}^u_{1:T}$. Given partial observations, we can perform inference using the observed features $\mathbf{x}^p_{1:T}$ only. Our approach learns to impute the unobserved features by extracting the relevant information therefor from the observation and action history and the learned model dynamics.

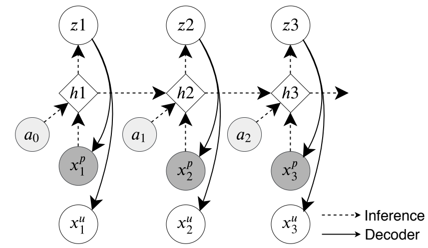


Figure 1: Our proposed partially observable sequential VAE. Shaded variables are observed.

Our key assumption is that learning to impute the unobserved features leads to better representations which can be leveraged by the task policy and that, because of partial observability, sequential representation learning is better as non-sequential learning. Furthermore, unlike many other sequential representation learning approaches for RL that only reason over the observation sequence $\mathbf{x}^p_{1:T}$, we take into account both $\mathbf{x}^p_{1:T}$ and the action sequence $\mathbf{a}_{1:T}$ for inference. The intuition is that since $\mathbf{x}^p_{1:T}$ by itself consists of limited information over the environment's underlying state, incorporating the action sequence provides additional information for inferring a belief state. To summarize, our approach learns to encode $\mathbf{x}^p_{1:T}$ and $\mathbf{a}_{1:T}$ into a latent representation 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 1.

Observation Decoder Let $\mathbf{z}_{1:T} = (\mathbf{z}_1, ..., \mathbf{z}_T)$ denote a sequence of latent states. We consider the probabilistic model $p_\theta(\mathbf{x}^p_{1:T}, \mathbf{x}^u_{1:T}, \mathbf{z}_{1:T}) = \prod_{t=1}^T p(\mathbf{x}^p_t, \mathbf{x}^u_t|\mathbf{z}_t)\, p(\mathbf{z}_t)$. For simplicity of notation, 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, $p(\mathbf{x}^p_t,\, \mathbf{x}^u_t|\mathbf{z}_t) = p(\mathbf{x}^p_t|\mathbf{z}_t)\, p(\mathbf{x}^u_t|\mathbf{z}_t)$.

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 1: $q_\phi(\mathbf{z}_{1:T}|\mathbf{x}_{1:T}, \mathbf{a}_{<T}) = \prod_{t=1}^T q_\phi(\mathbf{z}_t|\mathbf{x}^p_{\leq t}, \mathbf{a}_{<t})$, 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 $q_\phi(\cdot)$ [6]. 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 $\mathbf{z}_t$. Because of the supervised learning task, the belief state can provide abundant information for the missing features.

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. Note that the pretraining consumes only restricted amounts of data (i.e., 2000 for our case) so that in practice the cost of collecting such data for developing our method is generally acceptable. Concretely, the pre-training task updates the parameters θ, ϕ by maximizing the following variational lower-bound [10, 11, 33]:

$$\log p(\mathbf{x}^p_{1:T}, \mathbf{x}^u_{1:T}) \geq \mathbb{E}_{q_\phi}\Big[\sum_t \log p_\theta(\mathbf{x}^p_t|\mathbf{z}_t) + \log p_\theta(\mathbf{x}^u_t|\mathbf{z}_t) - \mathrm{KL}\big(q_\phi(\mathbf{z}_t|\mathbf{x}^p_{\leq t}, \mathbf{a}_{<t})\,||\,p(\mathbf{z}_t)\big)\Big] \quad (2)$$

By incorporating the term $\log p_\theta(\mathbf{x}^u_t|\mathbf{z}_t)$, training of the 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. Given the pre-trained representation learning model, the policy is trained in a multi-stage reinforcement learning setting, where the representation provided by sequential VAE is taken as input to the policy. Pseudocode for our algorithm is in the Appendix.

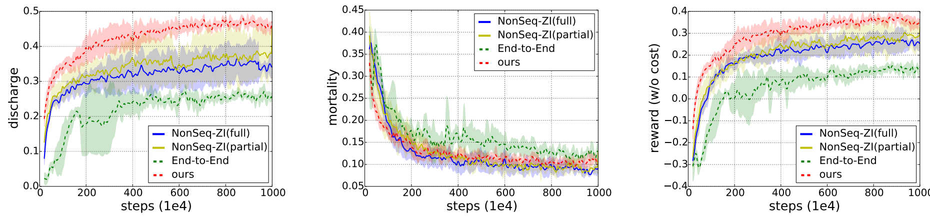


Figure 2: Performance curves in terms of discharge rate, mortality rate and reward (w/o cost) for the compared approaches on *Sepsis*. The curves are derived under cost value of 0.01. Our method converges to treatment policy with substantially better reward compared to the baselines.

3 Experiments

We evaluate our proposed approach in two experimental domains: a *sepsis* medical simulator fitted from real-world data [21] (further experiments are provided in the appendix); a *bouncing ball*$^+$ control task with high-dimensional image pixels as input, adapted from [4] (provided in the Appendix).

Baselines For comparison, we mainly consider variants of the strong VAE baseline *beta-VAE* [7], which works on non-time-dependent data instances. For representing the missing features, we adopt the *zero-imputing* method, proposed in [20] 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 [15]. We also consider an *end-to-end* baseline which does not employ pre-trained representation learning model. 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 Pre-training the VAE models requires data that enables to incorporate abundant dynamics information. Therefore, we collect a small scale dataset of 2000 trajectories, where half of the data is collected from a random policy and the other half from a policy which better captures the states that would be encountered by a learned model (e.g., by a data collection policy trained end-to-end or using human generated trajectories). Details are provided in the Appendix.

3.1 Sepsis Medical Simulator

Task Settings We adopt a medical simulator for treating sepsis in ICU patients [21]. The task is to learn to apply three *treatments* (*antibiotic*, *ventilation*, *vasopressors*). The state space consists of 8 features: 3 of them indicate the current *treatment* state; 4 of them are the *measurement* states (*heart rate*, *sysBP rate*, *percoxyg state*, *glucose level*). The 8th feature specifies 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.

Policy Training Results We show the policy training results for *Sepsis* in Figure 2. Overall, our proposed method results in substantially better task reward compared to the baselines. Note that the performance of discharge rate for our method increases significantly faster than baseline approaches, which shows that the model can quickly learn to apply appropriate treatment actions and thus be trained in a much more sample efficient way. Moreover, our method also converges to substantially better values than the baselines. Upon convergence, it outperforms the best non-sequential VAE baseline with a gap of $> 5\%$ for discharge rate. 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 is crucial to determine the effect of agent's policy training practice. The result also reveals that learning to impute missing features contributes greatly to improve the policy training performance.

4 Conclusion

We presented the novel AFA-POMDP framework where the task policy and the active feature acquisition policy are learned under a unified formalism. Our method incorporates a model-based representation learning attempt, where a sequential VAE model is trained to impute missing features via learning model dynamics and thus offer high quality representations to facilitate the joint policy training under 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.

References

[1] A. 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.

[2] A. Das, S. Kottur, J. M. Moura, S. Lee, and D. Batra. Learning cooperative visual dialog agents with deep reinforcement learning. In *Proceedings of the IEEE international conference on computer vision*, pages 2951–2960, 2017.

[3] C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pages 49–58, 2016.

[4] M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Advances in Neural Information Processing Systems*, pages 3601–3610, 2017.

[5] W. Gong, S. Tschiatschek, S. Nowozin, R. E. Turner, J. M. Hernández-Lobato, and C. Zhang. Icebreaker: Element-wise efficient information acquisition with a bayesian deep latent gaussian model. In *Annual Conference on Neural Information Processing Systems*, pages 14791–14802, 2019.

[6] K. Gregor, G. Papamakarios, F. Besse, L. Buesing, and T. Weber. Temporal difference variational auto-encoder. *ICLR*, 2019.

[7] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2016.

[8] I. Higgins, A. Pal, A. Rusu, L. Matthey, C. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner. Darla: Improving zero-shot transfer in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1480–1490, 2017.

[9] Z. Jie, X. Liang, J. Feng, X. Jin, W. Lu, and S. Yan. Tree-structured reinforcement learning for sequential object localization. In *Advances in Neural Information Processing Systems*, pages 127–135, 2016.

[10] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[11] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[12] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *ICLR*, 2016.

[13] Y. Ling, S. A. Hasan, V. Datla, A. Qadir, K. Lee, J. Liu, and O. 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)*, pages 895–905, 2017.

[14] C. Ma, S. Tschiatschek, J. M. Hernández-Lobato, R. Turner, and C. Zhang. Vaem: a deep generative model for heterogeneous mixed type data. *arXiv preprint arXiv:2006.11941*, 2020.

[15] C. Ma, S. Tschiatschek, K. Palla, J. M. Hernández-Lobato, S. Nowozin, and C. Zhang. EDDI: efficient dynamic discovery of high-value information with partial VAE. In *Proceedings of the 36th International Conference on Machine Learning*, pages 4234–4243, 2019.

[16] S. Mathe, A. Pirinen, and C. Sminchisescu. Reinforcement learning for visual object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2894–2902, 2016.

[17] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou,

H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[20] A. Nazabal, P. M. Olmos, Z. Ghahramani, and I. Valera. Handling incomplete heterogeneous data using vaes. *arXiv preprint arXiv:1807.03653*, 2018.

[21] M. Oberst and D. Sontag. Counterfactual off-policy evaluation with gumbel-max structural causal models. In *ICML*, 2019.

[22] Y.-S. Peng, K.-F. Tang, H.-T. Lin, and E. Chang. Refuel: Exploring sparse features in deep reinforcement learning for fast disease diagnosis. In *Advances in Neural Information Processing Systems*, pages 7322–7331, 2018.

[23] Y. Satsangi, S. Lim, S. Whiteson, F. Oliehoek, and M. White. Maximizing information gain in partially observable environments via prediction reward. *AAMAS*, 2020.

[24] H. Shim, S. J. Hwang, and E. Yang. Joint active feature acquisition and classification with variable-size set encoding. In *Advances in Neural Information Processing Systems*, pages 1368–1378, 2018.

[25] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[26] M. T. Spaan and P. U. Lima. A decision-theoretic approach to dynamic sensor selection in camera networks. In *ICAPS*, 2009.

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

[28] G. Vezzani, A. Gupta, L. Natale, and P. Abbeel. Learning latent state representation for speeding up exploration. *arXiv preprint arXiv:1905.12621*, 2019.

[29] R. A. Yeh, C. Chen, T. Yian Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do. Semantic image inpainting with deep generative models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5485–5493, 2017.

[30] J. Yoon, J. Jordon, and M. van der Schaar. Invase: Instance-wise variable selection using neural networks. In *ICLR*, 2018.

[31] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Y. Choi. Action-driven visual object tracking with deep reinforcement learning. *IEEE transactions on neural networks and learning systems*, 29(6):2239–2252, 2018.

[32] S. Zannone, J. M. Hernández-Lobato, C. Zhang, and K. Palla. Odin: Optimal discovery of high-value information using model-based deep reinforcement learning. In *ICML Real-world Sequential Decision Making Workshop*, 2019.

[33] C. Zhang, J. Bütepage, H. Kjellström, and S. Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

[34] M. Zhang, S. Vikram, L. Smith, P. Abbeel, M. J. Johnson, and S. Levine. Solar: Deep structured latent representations for model-based reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[35] Z. Zheng and L. Sun. Disentangling latent space for vae by label relevant/irrelevant dimensions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12192–12201, 2019.

Appendix

This supplementary material is organized as follows. First, we present further related work and the pseudocode for our algorithm. Then we present additional experiment details on the *BouncingBall+* task and the *Sepsis* task. For each task, we present the task specifications, implementation details and additional evaluation results. Furthermore, we present a case study that investigates the efficiency of our proposed sequential representation model when trained with data under different levels of observability.

A Related Work

Our work jointly considers active learning and 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.

Active Feature Acquisition Our work draws motivation from the 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 at each time. One typical example is the conventional linear model that poses sparsity inducing prior distribution to the model [27]. There is an alternative category that models feature acquisition as a Bayesian experiment design [5, 14, 15]. However, the sequential decision making is for variable-wise feature acquisition and the problems are still non time-series tasks in nature. There are also a number of approaches that adopt reinforcement learning to actively find optimal feature subsets, with successful applications in various research fields, such as active perception/sensor selection [26, 23], visual object localization/tracking [31, 9] and medical diagnosis [30, 32]. Most of those works focus on learning a policy for active feature acquisition only, whereas we consider a problem of simultaneously learning a reinforcement learning policy and an active feature acquisition policy. Besides our primary focus on dealing with time series data, the problem we consider is also settled on more complicated system dynamics than the aforementioned works, as performing feature acquisition would greatly reduce the degree of observability for agent when learning task skills and thus makes it more challenging to learn optimal task skills.

Representation Learning in POMDP Learning reinforcement learning policies with active feature acquisition results in a policy training scenario with partial observability, for which learning meaningful representation would become an essential and non-trivial research challenge. Most conventional approaches unifies the process of representation learning with policy training and results in policies trained in an end-to-end fashion [12, 17, 18]. However, such models often engage trainable parameters with considerable size and result to be less sample efficient. Another strand of works tackles the representation learning for POMDP in an off-line fashion, which results in multi-stage reinforcement learning. In [7, 8], pretrained VAE models are adopted as the representation module to build agents with strong domain adaptation performance. The key difference between their works and ours is in that they consider typical POMDP domains where the state presents partial view over the environment and they propose a non-sequential VAE model, whereas ours considers a setting where feature-level information could be *missing* and we propose a sequential representation learning approach to infer a more informative state representation. Recently, there emerged a fruitful literature over sequential representation learning for POMDP [6, 28], where most of them formulate VAE training as an auxiliary task for policy training. In our work, we consider a model-based representation learning attempt, where a sequential generative model is trained to learn model dynamics and generate high-quality features. Our attempt of learning model dynamics to gather information over the unobserved features is also related to image inpainting works to a certain extent [29, 35]. However, such methods mostly focus on inpainting static images, such as face images, whereas we consider imputing the features from time-series data. Apart from this, our primary focus is on learning reinforcement learning policies with active feature acquisition, rather than considering image inpainting only.

B Pseudocode of our Algorithm

Algorithm 1 RL with Active Feature Acquisition

```
Input: learning rate α > 0, dataset 𝒟
Initialize RL policy π_f, π_c, VAE parameters θ, φ.
Train VAE on dataset 𝒟 using Eq (2).
while Not Converge do
    Reset the environment.
    Initialize null observation x_1^p = , feature acquisition action a_0^f and control action a_0^c.
    for i = 1 to T do
        Compute representation with VAE: b_t = q_φ(x_{≤t}^p, a_{<t}).
        Sample a feature acquisition action a_t^f ~ π_f(b_t) and a control action a_t^c ~ π_c(b_t).
        Step the environment and receive partial features, reward and terminal: x_{t+1}^p, r_t, term ~ env(a_t^f, a_t^c)
        Compute cost c_t = Σ_i c · 𝕀(a_t^{f(i)}).
        Save the transitions {b_t, a_t^f, a_t^c, r_t, c_t, 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
```

C Bouncing Ball$^+$

C.1 Task Specifications

We adapted the original *bouncing ball* experiment presented in [4]. 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 the agent reaches the target location.

C.2 Implementation Details

For all the baseline methods, *Zero-Imputing* [20] 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 *deconvolutional* 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 computed for the observed features; the other employs the *full* loss that is computed for all the features, 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 1.

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 size 32 to generate a 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 1.

Table 1: Hyperparameter settings for training VAE models on the *Bouncing Ball*$^+$ dataset.

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

LSTM-A3C We adopt LSTM-A3C [17] 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 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 an end-to-end method. To train the end-to-end method, we employ a cost of 0.01 over the 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 2. We notice that our proposed method, by employing an advanced representation model, leads to a significantly better feature acquisition policy than *End-to-End* (smaller number of observations while achieving similar or better reward).

C.4 First Set of Experiments

Representation Learning Results We evaluate the missing feature imputing performance of each VAE model in terms of negative log likelihood (NLL) and present results in Table 3. We notice that our proposed model yields a significantly better imputing result than all the other baselines. This demonstrates that our proposed sequential VAE model can efficiently capture the environment dynamics and learn meaningful information over the missing features. Such efficiency is vital in determining both the acquisition and task policy training performance in AFA-POMDP, since

Table 2: Task performance for the data collection policy and our proposed method on *Bouncing Ball*$^+$.

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

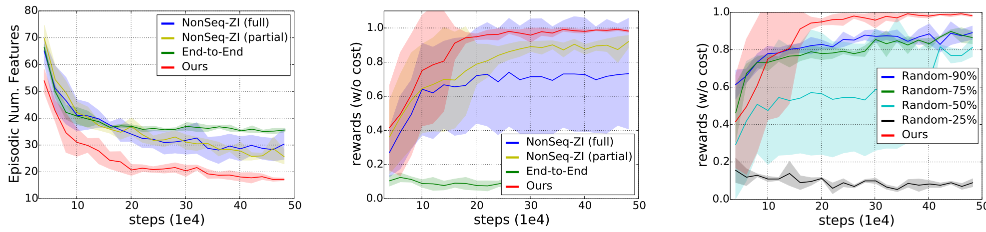


Figure 3: Performance curves on the *bouncing ball*$^+$ domain: **a**: episodic number of observations acquired by the π^f; **b**: task rewards w/o cost. Our proposed method outperforms the non-sequential baselines in learning the task as well as acquiring less observations; **c**: Ablation study on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

Table 3: Missing feature imputing loss evaluated on *Bouncing Ball*$^+$ and *Sepsis*.

VAE MODEL	BOUNCING BALL$^+$ (NLL)	SEPSIS (MSE)
NONSEQ-ZI (PARTIAL)	0.6504 ($\pm$ 0.1391)	0.8441 ($\pm$0.0586)
NONSEQ-ZI (FULL)	0.0722 ($\pm$ 0.0004)	0.4839 ($\pm$ 0.0012)
SEQ-PO-VAE (OURS)	**0.0324** ($\pm$ 0.0082)	**0.1832** ($\pm$0.0158)

both policies are conditioned on the VAE latent features. We also demonstrate sample trajectories reconstructed by different VAE models in Appendix. The results show that our model learns to impute considerable amount of missing information.

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 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 plays a vital role in 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 accomplish the task.

We show that our proposed method can learn meaningful feature acquisition policies. We visualize 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 Figure 3 (c), our method converges to better standard than random policies with considerably high selection probabilities.

C.5 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

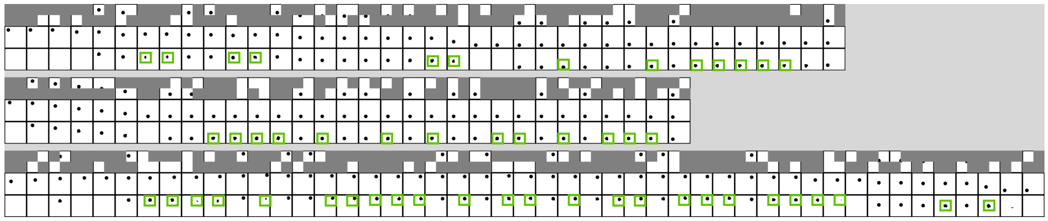


Figure 4: *Seq-PO-VAE* reconstruction for the online trajectories upon convergence (better to view enlarged). Each block of three rows corresponds to the results for one trajectory. In each block, the three rows (top-down) correspond to: (1) the partially observable input selected by acquisition policy; (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).

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 5, 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.

C.6 Investigation on Cost-Performance Trade-off

We perform a case study on investigating the cost-performance trade-off for each representation learning method and present the results in Figure 6. Apparently, as we increase the cost, the exploration-exploitation task becomes more challenging and each compared method has its *own upper limit of cost*, above which the model would fail to learn an effective task policy while acquiring minimum observations. First, we notice that the *End-to-End* model takes a long time to progress in learning task skills (i.e., typically > 1.5m), while the VAE-based models can progress much faster. Among the VAE-based methods, we notice that our proposed method (Figure 6(d)) can accomplish the task by acquiring as little as 8 observations whereas the baselines *NonSeq-ZI (Full)* (Figure 6(b)) and *NonSeq-ZI (partial)* (Figure 6(c)) achieve a standard of acquiring approximately 20 observations (refer to the lowest point among the *solid* lines in the figure). Thus, we conclude that our proposed approach can significantly benefit the cost-sensitive policy training and leads to a policy which acquires fewer observations while achieving equal or better task performance.

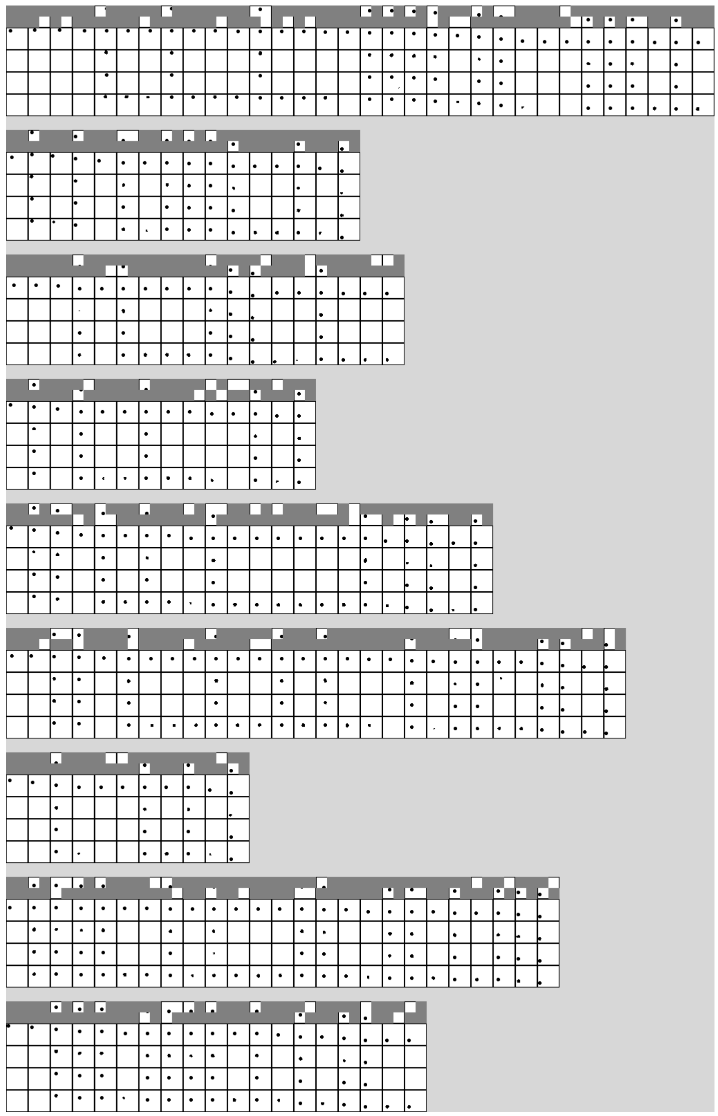
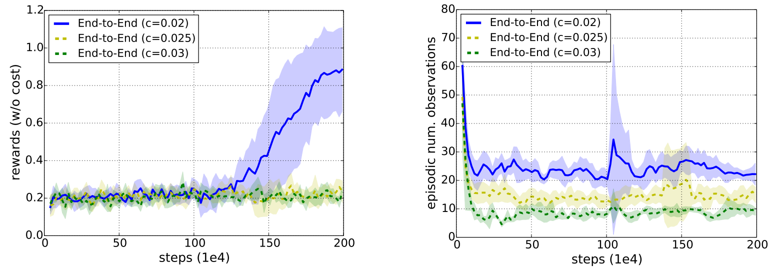
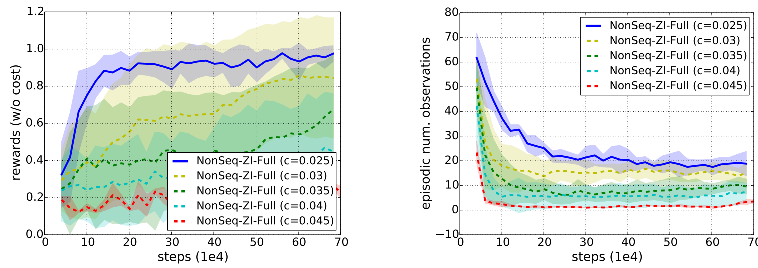


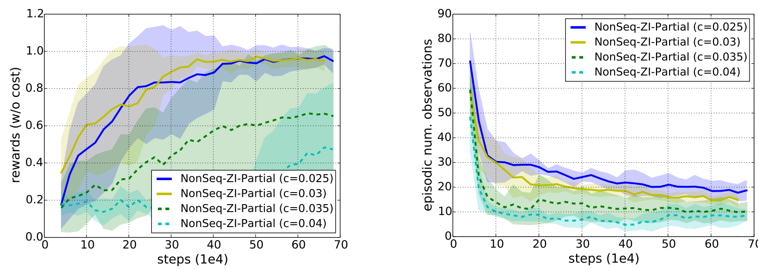
Figure 5: Imputation results for different VAE models. We select 9 trajectories obtained from the trained *End-to-End* policy. Each block corresponds to the results for one trajectory (better to view enlarged). The five rows in one block are (top-down): (1) partial observations acquired by the agent; (2) ground-truth image with full observation; (3) Imputation by *NonSeq-ZI (partial)*; (4) Imputation by *NonSeq-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.



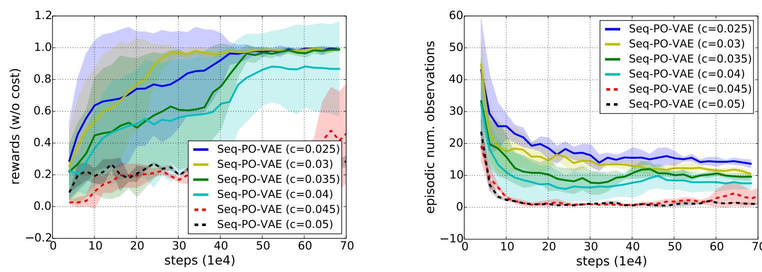
(a) End-to-End



(b) NonSeq-ZI (full)



(c) NonSeq-ZI (partial)



(d) Seq-PO-VAE (ours)

Figure 6: Cost-performance trade-off investigation. Each row corresponds to the performance in terms of task reward (left) and number of acquisitions (per episode) obtained for a specific method (right), for a specific method (see the legend). Each curve is derived from 10 independent runs. We use dotted lines to indicate those instances for which 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. Seq-PO-VAE consumes less than 10 observations to accomplish this task.

D Sepsis Medical Simulator

D.1 Task Specifications

For this task we employ a Sepsis simulator proposed in previous work [21]. 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 an active feature acquisition strategy is to help the decision making system to reduce *measurement* cost during its execution.

D.2 Implementation Details

For all the compared methods, we adopt *Zero-Imputing* [20] 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 function.

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.

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 the identical architecture as that for *NonSeq-ZI*. The details for training *Seq-PO-VAE* are presented in Table 4.

LSTM-A3C The LSTM-A3C [17] takes encoded state features derived from the corresponding representation model as its input. The encoded features are 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.

D.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 *End-to-End* with cost 0.0. All the VAE models are evaluated on a test dataset that consists of identical size

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

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

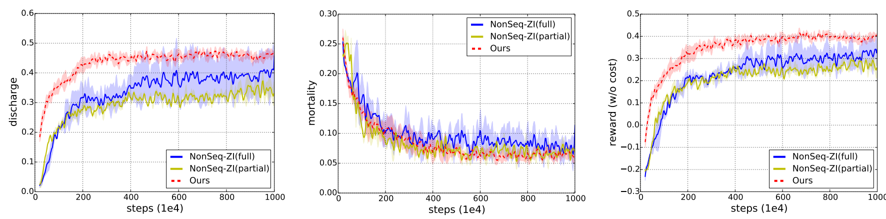
and data distribution as the the training dataset. We present the task treatment reward obtained by our data collection policy derived from the *End-to-End* method and that obtained by our proposed method in Table 5. Noticeably, by performing representation learning, our method could obtain much better treatment reward compared to the data collection policy. Therefore, it is essential to conduct representation learning to tackle the challenging AFA-POMDP problem.

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

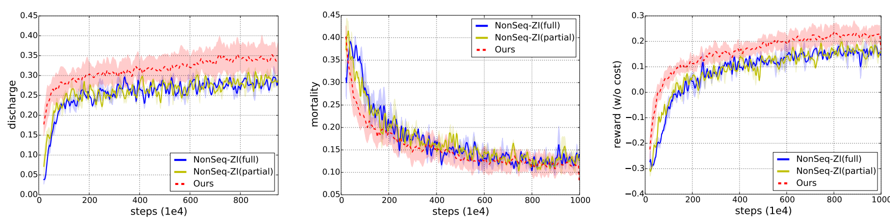
	Model	
	End-to-End	Ours
Treatment Reward	0.35	**0.45**

D.4 More Comparison Result under Different Values for Cost

We present the cost-performance trade-off on *Sepsis* domain when running our method under different cost values in $\{0, 0.025\}$. The results are shown in Figure 7(a) and Figure 7(b)). By increasing the value of cost, we obtain a feature acquisition policy that acquires substantially less features within each episode, with a sacrifice in task rewards.



(a) Cost=0



(b) Cost=0.025

Figure 7: Comparison result between our proposed method and the non-sequential VAE baseline models under different values for cost.

Furthermore, we present the episodic number of acquired features for our method in Figure 8) when trained under different cost values. The results show that by increasing the cost, the number of feature acquisition substantially reduces.

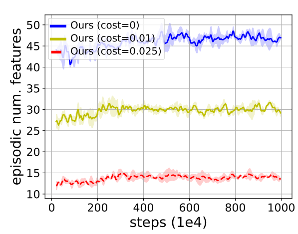


Figure 8: Average num. observations acquired in each episode under cost values in $\{0, 0.1, 0.025\}$.

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

We present two illustrative examples in Figure 9 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 can 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 9, 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 can be greatly beneficial for training the task policy and feature acquisition policy.

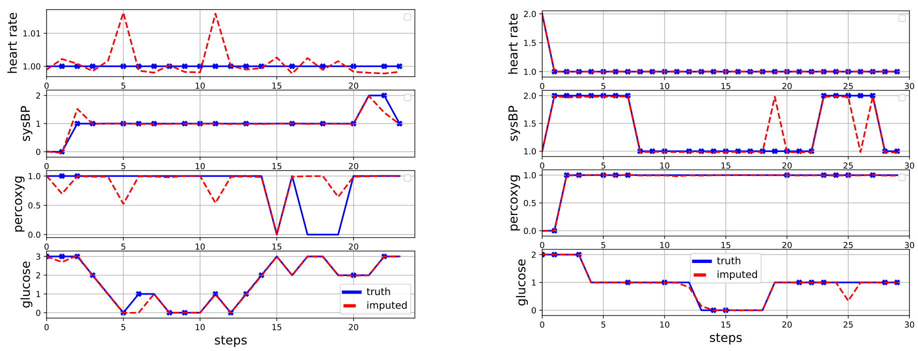


Figure 9: Two example trajectories for illustrating how our method works on the *Sepsis* medical domain. The acquisition policy is trained with a cost of 0. Each block corresponds to one trajectory and the four rows correspond to the four *measurement* features being considered for active feature acquisition. 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 by giving reasonable imputation over the unobserved features.

D.6 Ablation Study

In this section, we present an ablation study on the *Sepsis* medical domain.

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 10. From the results, we notice that there is a clear cost-performance trade-off, i.e., a 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, our method acquire only about half of the total number of features (refer to the x-value derived by *Random*-100%) to obtain comparable task performance. We also notice that the number of features acquisition decreases significantly as the cost increases. Therefore, our proposed framework can be applied to obtain feature acquisition policies that meet different levels of budget.

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 11. 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. Overall, conducting policy training for AFA-POMDP with our proposed representation learning method could lead to subsequent reduce in total feature acquisition cost compared to the baseline methods.

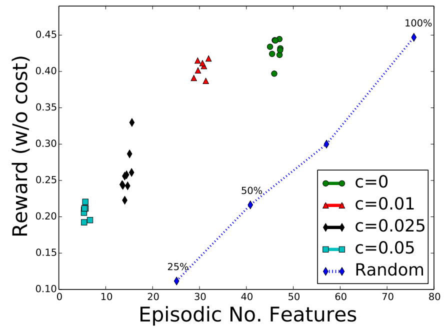


Figure 10: Comparison between active feature acquisition (performed under different cost values) vs. random feature acquisition. The results are obtained from *Sepsis* domain.

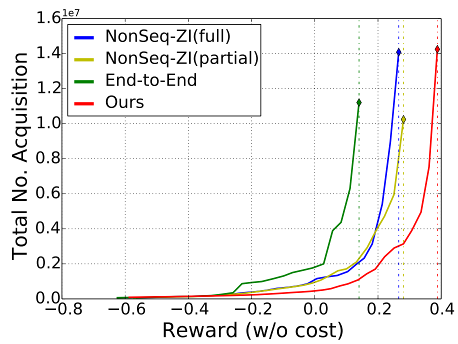


Figure 11: Total feature acquisition cost consumed by different approaches to obtain task performance (i.e., reward) at certain standards. The results are obtained from *Sepsis* domain.

E Case Study: Investigating the Data Observability for Representation Learning

In our proposed method, we assumed that the model has access to the fully observed data at the representation learning stage, so that the VAE can be trained to impute the missing features with the supervision of the fully observed data (following Equation (5) in the paper). In this section, we present a case study to demonstrate that such assumption does not necessarily need to hold and that our method can work with partially observed training data as well. To this end, we create two adapted baselines from our proposed method, where the representation learning models (i.e., Seq-PO-VAE) for the baselines are trained under partial observation, i.e., only 50%/90% of the features are accessible when training the Seq-PO-VAE model where the features to observe are randomly selected. We denote such adapted baselines as *Seq-PO-VAE (50%)* and *Seq-PO-VAE (90%)*, respectively.

We present the missing feature imputing performance for the VAE models evaluated on the two task domains in Table 6. From the results, we notice that with reduced observability, the missing feature imputing performance for *Seq-PO-VAE (50%/90%)* degrades to fall below *Seq-PO-VAE (full)*, which is as expected. However, the adapted baselines with partial observability can still benefit from our proposed sequential modeling with dynamics learning a lot. As a result, *Seq-PO-VAE (50%/90%)* can outperform the non-sequential baselines *NonSeq-ZI (partial/full)* on both missing feature imputing tasks with substantial performance margins. Note that the model *NonSeq-ZI (full)* still employs full observation over the dataset during its training, but its missing feature imputing performance is substantially inferior as compared to *Seq-PO-VAE (50%)*. Overall, the above results demonstrate that our proposed representation learning method can derive meaningful representation with considerable efficiency in imputing missing features even when the model is trained under partial observation.

Furthermore, we demonstrate the policy training performance for the *Seq-PO-VAE (50%/90%)* baselines evaluated on the *Sepsis* domain. The results are shown in Figure 12. As expected, the performance of *Seq-PO-VAE* trained with partial observation degrades from that trained with full observation. The reason is due to that the task of imputing the missing features via learning system

Table 6: Missing feature imputing loss evaluated on *Bouncing Ball*$^{+}$ and *Sepsis* domains.

VAE model	**Bouncing Ball**$^{+}$ (NLL)	**Sepsis** (MSE)
NonSeq-ZI (partial)	0.6504 (± 0.1391)	0.8441 (±0.0586)
NonSeq-ZI (full)	0.0722 (± 0.0004)	0.4839 (± 0.0012)
Seq-PO-VAE (50%)	0.0375 (± 0.0010)	0.2892 (± 0.0097)
Seq-PO-VAE (90%)	0.0381 (± 0.0015)	0.2450 (± 0.0096)
Seq-PO-VAE (full)	**0.0324** (± 0.0082)	**0.1832** (±0.0158)

dynamics could be extremely challenging when only partial features are presented during training. However, when the level of observability is high, the model can still lead to promising performance that outperforms the non-sequential VAE baselines. Overall, the results reveal that our proposed method works best with full observability, but it is promising to work with partial observability when the level of observability is relatively high. Adapting our proposed method to tackle challenging AFA-POMDP domains with restricted level of observability to data is subject to future work, and our approach will benefit from any advances in representation learning from partially observed data.

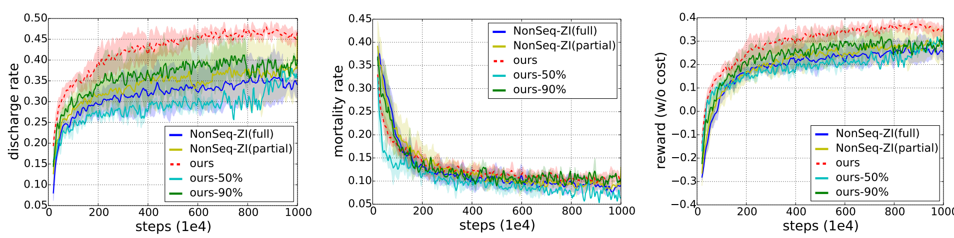


Figure 12: Performance curves in terms of *discharge rate*, *mortality rate* and *reward (w/o cost)* on *Sepsis* domain, evaluated with a cost value of 0.01.