Continuous State-Space Models for Optimal Sepsis Treatment: a Deep Reinforcement Learning Approach

Aniruddh Raghu [1,2] ARAGHU@MIT.EDU

Matthieu Komorowski [3] MKOMO@MIT.EDU

Leo Anthony Celi [3] LCELI@MIT.EDU

Peter Szolovits [1] PSZ@MIT.EDU

Marzyeh Ghassemi [1] MGHASSEM@MIT.EDU

1. Computer Science and Artificial Intelligence Lab, MIT, Cambridge, MA
2. Department of Engineering, University of Cambridge, UK
3. Laboratory for Computational Physiology, MIT, Cambridge, MA

Abstract

Sepsis is a leading cause of mortality in intensive care units (ICUs) and costs hospitals billions annually. Treating a septic patient is highly challenging, because individual patients respond very differently to medical interventions and there is no universally agreed-upon treatment for sepsis. Understanding more about a patient's physiological state at a given time could hold the key to effective treatment policies. In this work, we propose a new approach to deduce optimal treatment policies for septic patients by using continuous state-space models and deep reinforcement learning. Learning treatment policies over continuous state-spaces is important, because doing so allows us to retain more of the patient's physiological information. Our model is able to learn clinically interpretable treatment policies, similar in important aspects to the treatment policies of physicians. Evaluating our algorithm on past ICU patient data, we find that our model could reduce absolute patient mortality in the hospital by up to 3.6% over observed clinical policies. The learned treatment policies could be used to aid intensive care clinicians in medical decision making and improve the likelihood of patient survival.

1. Introduction

Sepsis (severe infections with organ failure) is a dangerous condition that costs hospitals billions of pounds in the UK alone (Vincent et al., 2006), and is a leading cause of patient mortality (Cohen et al., 2006). The clinicians' task of deciding treatment type and dosage for individual patients is highly challenging. Besides antibiotics and infection control, a cornerstone in managing severe infections is the administration of intravenous fluids to correct hypovolemia (a state in which the blood plasma is too low). This may be followed by the administration of vasopressors to counteract sepsis-induced vasodilation (the dilation of blood vessels resulting in reduced blood pressure).

Using various fluids and vasopressors treatment strategies have been shown to lead to extreme variations in patient mortality, which demonstrates how critical these decisions are (Waechter et al., 2014). While international efforts attempt to provide general guidance for treating sepsis, physicians at the bedside still lack efficient tools to provide individualized real-time decision support (Rhodes et al., 2017). As a consequence, individual clinicians vary treatment in many ways, e.g., the amount and type of fluids used, the timing and dosing of vasopressors given, which antibiotics are given, and whether to administer corticosteroids.

In this work, we propose a data-driven approach to discover optimal sepsis treatment strategies. We use deep reinforcement learning (RL) algorithms to identify how best to treat septic patients in the intensive care unit (ICU) to improve their chances of survival. While RL has been used successfully in complex decision making tasks (Mnih et al., 2015; Silver et al., 2016), its application to clinical models has thus far been limited by data availability (Nemati et al., 2016) and the inherent difficulty of defining clinical state and action-spaces (Prasad et al., 2017; Komorowski et al., 2016).

Nevertheless, RL algorithms have many desirable properties for the problem of deducing high-quality treatments. Their intrinsic design for sparse reward signals makes them well suited to overcome complexity from the stochasticity in patient responses to medical interventions, and delayed indications of treatment efficacy. Importantly, RL algorithms also allow us to infer optimal strategies from suboptimal training examples.

In this work, we demonstrate how to surmount the modeling challenges present in the medical environment and use RL to deduce optimal treatment policies for septic patients.[1] We focus on continuous state-space modeling, representing a patient's physiological state at a point in time as a continuous vector (using either raw physiological data or sparse latent state representations), and find optimal actions with Deep Q-Learning (Mnih et al., 2015). Motivating this approach is the fact that physiological data collected from ICU patients provide very rich representations of a patient's physical state, allowing for the discovery of interpretable and high-quality policies.

In particular, we:

1. Propose deep reinforcement learning models with continuous state-spaces, improving on earlier work with discrete state-spaces.
2. Identify treatment policies that could improve patient outcomes, potentially reducing absolute patient mortality in the hospital by 1.8 - 3.6%, from a baseline absolute mortality of 13.7%.
3. Investigate the learned policies for clinical interpretability and potential use as a clinical decision support tool.

2. Background and Related Work

In this section we outline important reinforcement learning algorithms used in the paper and motivate our approach in comparison to prior work.

2.1 Reinforcement Learning

Reinforcement learning (RL) models time-varying state-spaces with a Markov Decision Process (MDP), in which at every timestep t an agent observes the current state of the environment s_t, takes an action a_t from the allowable set of actions $\mathcal{A} = \{1, \ldots, M\}$, receives a reward r_t, and then transitions to a new state s_{t+1}. The agent selects actions at each timestep that maximize its expected

1. Either patients who develop sepsis in their ICU stay, or those who are already septic at the start of their stay.

discounted future reward, or *return*, defined as $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$, where γ captures the tradeoff between immediate and future rewards, and T is the terminal timestep. The optimal action value function $Q^*(s, a)$ is the maximum discounted expected reward obtained after executing action a in state s; that is, performing a in state s and proceeding optimally from this point onwards. More concretely, $Q^*(s, a) = \max_\pi \mathbb{E}[R_t | s_t = s, a_t = a, \pi]$, where π — also known as the *policy* — is a mapping from states to actions. The optimal value function is defined as $V^*(s) = \max_\pi \mathbb{E}[R_t | s_t = s, \pi]$, where we act according to π throughout.

In Q-learning, the optimal action value function is estimated using the Bellman equation, $Q^*(s, a) = \mathbb{E}_{s' \sim T(s'|s,a)}[r + \gamma \max_{a'} Q^*(s', a') | s_t = s, a_t = a]$, where $T(s'|s, a)$ refers to the state transition distribution. Learning proceeds either with value iteration (Sutton and Barto, 1998) or by directly approximating $Q^*(s, a)$ using a function approximator (such as a neural network) and learning via stochastic gradient descent. Note that Q-learning is an *off-policy* algorithm, as the optimal action-value function is learned with samples $< s, a, r, s' >$ that are generated to explore the state-space. An alternative to Q-learning is the SARSA algorithm (Rummery and Niranjan, 1994); an on-policy method to learn $Q^\pi(s, a)$, which is the action-value function when taking action a in state s at time t, and then proceeding according to policy π afterwards.

In this work, the state s_t is a patient's physiological state, either in raw form (Section 3.2) or as a latent representation (Section 4.3). The action-space, $\mathcal{A}$, is of size 25 and is discretized over doses of vasopressors and IV fluids, two drugs commonly given to septic patients (Section 3.3). The reward r_t is $\pm R_{max}$ at terminal timesteps and zero otherwise, with positive rewards being issued when a patient survives, and negative rewards when a patient dies. At every timestep, the agent is trained to take an action a_t with the highest Q-value, aiming to increase the chance of patient survival.

2.2 Reinforcement Learning in Health

Much prior work in clinical machine learning has focused on supervised learning techniques for diagnosis (Esteva et al., 2017) and risk stratification (N.Razavian et al., 2015). The incorporation of time in a supervised setting could be implicit within the feature space construction (Hug and Szolovits, 2009; Joshi and Szolovits, 2012), or captured with multiple models for different time points (Fialho et al., 2013; Ghassemi et al., 2014). We prefer RL for sepsis treatment over supervised learning because the ground truth of "good" treatment strategy is unclear in medical literature (Marik, 2015).

Nemati et al. (2016) applied deep RL techniques to model ICU heparin dosing as a Partially Observed Markov Decision Process (POMDP), using both discriminative Hidden Markov Models and Q-networks to discover the optimal policy. Their investigation was made more challenging by the relatively small amount of available data. Shortreed et al. (2011) learned optimal treatment policies for schizophrenic patients, and quantified the uncertainty around the expected outcome for patients who followed the policies. Prasad et al. (2017) use off-policy reinforcement learning algorithms to determine ICU strategies for mechanical ventilation administration and weaning, but focus on simpler learning algorithms and a heuristic action-space. In contrast, we experiment with using a sparse autoencoder to generate latent representations of the state of a patient, likely leading to an easier learning problem. We also propose neural network architectures that obtain more robust methods for optimal policy deduction.

While little prior work exists, Komorowski et al. (2016) use a discretized state and action-space to deduce optimal treatment policies for septic patients. Their work applied SARSA learning to fit

an action-value function to the physician policy and value-iteration techniques to find an optimal policy (Sutton and Barto, 1998). The optimal policy was then evaluated by comparing the Q-values that would have been obtained following chosen actions to the Q-values obtained by the physicians. We reproduce a similar model as our baseline, using related data pre-processing and clustering techniques. Notably, we differ from Komorowski et al. (2016) in the following ways: we focus on continuous modeling, where policies are learned directly from the physiological state data, without discretization; we propose a novel evaluation metric (Jiang and Li, 2015); and we focus on in-hospital mortality instead of 90-day mortality because of the other unobserved factors that could affect mortality in a 3-month timeframe.

3. Data and Preprocessing

3.1 Cohort

Data for these patients were obtained from the Multiparameter Intelligent Monitoring in Intensive Care (MIMIC-III v1.4) database (Johnson et al., 2016), which is publicly available and contains hospital admissions from over 38,600 adults (at least 15 years old). We extracted a cohort of patients fulfilling the Sepsis-3 criteria (Singer et al., 2016), and note that summary information about the populations is similar in sepsis survivors and non-survivors (Table 1).

	% Female	Mean Age	Hours in ICU	Total Population
Survivors	43.6	63.4	57.6	15,583
Non-survivors	47.0	69.9	58.8	2,315

Table 1: Comparison of cohort statistics for subjects that fulfilled the Sepsis-3 criteria.

3.2 Feature Preprocessing

For each patient, we extracted relevant physiological parameters including demographics, lab values, vital signs, and intake/output events. Data were aggregated into windows of 4 hours, with the mean or sum being recorded (as appropriate) when several data points were present in one window. Variables with excessive missingness were removed, and any remaining missing values were imputed with k-nearest neighbors, yielding a 47×1 feature vector for each patient at each timestep. Values exceeding clinical limits were capped, and capped data were normalized per-feature to zero mean and unit variance. See Appendix 8.3 for a full feature list.

3.3 Action Discretization

We define a 5×5 action-space for the medical interventions, representing the volume of intravenous (IV) fluid (adjusted for fluid tonicity) and maximum vasopressor (VP) dosage given in a 4 hour window. The action-space was restricted to these two interventions as both drugs are extremely important in the management of septic patients, but there is no agreement on when, and how much, of each drug to give (Marik, 2015). We discretized the action-space into per-drug quartiles based on all non-zero dosages of the two drugs, and converted each drug at every timestep into an integer representing its quartile bin. We included a special case of no drug given as bin 0. This created an action representation of interventions as tuples of (total IV in, max VP in) at each time.

4. Methods

The challenge of applying RL to optimal medication dosing is that all available data are *offline sampled*; that is, data are collected previously and models can only be fit to a retrospective dataset. In an RL context, this limits exploration of the state-space in question, and makes learning the truly 'optimal' policy difficult. This limitation motivates trying several different approaches, with varied modeling constraints, to determine the best medication strategy for patients.

We focus on off-policy RL algorithms that learn an optimal policy through data that are generated by following an alternative policy. This makes sense for our problem because the available data are generated from a policy followed by physicians, but our goal is to learn a different, optimal policy rather than to evaluate the physician's policy. We propose deep models with continuous state-spaces and discretized action-spaces to retain more of the underlying state representation.

4.1 Discretized State-space and Discretized Action-space

Following Komorowski et al. (2016), we create a baseline model with discretized state and action-spaces, aiming to capture the underlying representation while simplifying the learning procedure. We use this approach to evaluate the performance of other techniques, and to understand the significance of learned Q-values. We also use the SARSA algorithm (Rummery and Niranjan, 1994) to learn $Q^{\pi}(s, a)$, and the action-value function for the physician policy (Appendix 8.4).

4.2 Continuous State-spaces

Continuous state-space models directly capture a patient's physiological state, and allow us to discover high-quality treatment policies. To learn an optimal policy with continuous state vectors, we use neural networks to approximate the optimal action-value function, $Q^*(s, a)$.

Our model is based on a variant of Deep Q-Networks (Mnih et al., 2015). Deep Q-Networks seek to minimize a squared error loss between the output of the network, $Q(s, a; \theta)$, and the desired target, $Q_{target} = r + \gamma \max_{a'} Q(s', a'; \theta)$, observing tuples of the form $< s, a, r, s' >$. The network has outputs for all the different actions that can be taken — for all $a \in \mathcal{A} = \{1, \ldots, M\}$. Concretely, the parameters θ^* are found such that:

$$\theta^* = \arg\min_{\theta} \mathbb{E}\left[\mathcal{L}(\theta)\right] = \arg\min_{\theta} \mathbb{E}\left[\left(Q_{target} - Q(s, a; \theta)\right)^2\right]$$

In practice, the expected loss is minimized via stochastic batch gradient descent. However, this method can be unstable due to non-stationarity of the target values, and using a separate network to determine the target Q-values ($Q(s', a')$), which is periodically updated towards the main network (used to estimate $Q(s, a)$), helps to improve performance.

Simple Q-Networks have several shortcomings, so we made important modifications to make our model suitable. Firstly, Q-values are frequently overestimated in practice, leading to incorrect predictions and poor policies. We solve this problem with a Double-Deep Q-Network (van Hasselt et al., 2015), where the target Q-values are determined using actions found through a feed-forward pass on the main network, as opposed to being determined directly from the target network. Secondly, in the context of finding optimal treatments, we want to separate the influence on Q-values of 1) a patient's *underlying state* being good (e.g. near discharge), and 2) the correct action being taken at that timestep. To this end, we use a Dueling Q-Network (Wang et al., 2015), where the action-value function $Q(s, a)$ is split into separate *value* and *advantage* streams, where the *value*

represents the quality of the current state, and the *advantage* represents the quality of the chosen action. Thirdly, training such a model can be slow as reward signals are sparse and only available on terminal timesteps. We use Prioritized Experience Replay (Schaul et al., 2015) to accelerate learning by sampling a transition from the training set with probability proportional to the previous error observed.

Our final network architecture is a Dueling Double-Deep Q-Network (Dueling DDQN), combining both of the above ideas. The network has two hidden layers of size 128, uses batch normalization (Ioffe and Szegedy, 2015) after each, Leaky-ReLU activation functions, a split into equally sized advantage and value streams, and a projection onto the action-space by combining these two streams (see Appendix 8.5). After training the Dueling DDQN, we can then obtain the optimal policy for a given patient state as: $\pi^*(s) = \arg\max_a Q(s, a)$.

4.3 Autoencoder Latent State Representation

Deep RL approaches for optimal medication are challenging to learn, because the patient state is a continuous vector without clear structure. We examined both ordinary autoencoders (Bengio, 2009) and sparse autoencoders (Ng, 2011) to produce latent state representations of the physiological state vectors and simplify the learning problem. Sparse autoencoders were trained with an additional term in the loss function to encourage sparsity (Section 8.6). Our autoencoder models all had a single hidden layer, which was used as the latent state representation. These latent state representations were used as inputs to the Dueling DDQN (Section 4.2).

5. Evaluation

The evaluation of off-policy models is challenging because it is difficult to estimate whether the rollout of a learned policy (using the learned policy to determine actions at each state) would eventually lead to lower patient mortality. Furthermore, directly comparing Q-values on off-policy data, as done in prior applications of RL to healthcare (Komorowski et al., 2016), can provide incorrect performance estimates (Jiang and Li, 2015). In this work, we propose evaluating learned policies with several approaches.

5.1 Discounted Returns vs. Mortality

To understand how expected discounted returns relate to mortality, we bin Q-values obtained via SARSA on the test set into discrete buckets, and for each, if it is part of a trajectory where a patient died, we assign it a label of 1; if the patient survived, we assign a label of 0. These labels represent the ground truth, as we know the actual outcome of patients when the physician's policy is followed. We compute the average mortality in each bin, enabling us to produce an empirically derived function of proportion of mortality versus expected return. We expect to see an inverse relationship between mortality and expected return, and this function enables us to associate returns with mortality for the purpose of evaluation.

5.2 Off-Policy Evaluation

We use the method of Doubly Robust Off-policy Value Evaluation (Jiang and Li, 2015) to evaluate policies. For each trajectory H we compute an unbiased estimate of the value of the learned policy, V_{DR}^{H}, and average the results obtained across the observed trajectories. We can also compute the

mean discounted return of chosen actions under the physician policy. Using both these estimates, and the empirically learned proportion of mortality vs. expected return function, we can assess the potential improvement our policy could bring in terms of reduction in patient mortality. Directly comparing the value (or return) of policies without the use of such an estimator is likely to give invalid results (Jiang and Li, 2015).

5.3 Qualitative Examination of Treatment Policies

We examine the overall choice of treatments proposed by the optimal policy to derive more clinical understanding, and compare these choices to those made by physicians to understand how differences in the chosen actions contribute to patient mortality.

6. Results

6.1 Fully Discretized Models are Well-calibrated with Test Set Mortality

Figure 1 shows the proportion of mortality versus the expected return for the physician policy on the held out test set. Note that $R_{max} = 15$ is the reward issued at terminal timesteps. As expected, we observe high mortality with low returns, and low mortality with high returns. The empirically derived mortality for the physician's policy matches the actual proportion of mortality in the test set. For the empirically derived mortality, we average the expected return for the physician on the test set to obtain $13.9 \pm 0.5\%$. This reflects the actual proportion of mortality on the test set (13.7%).

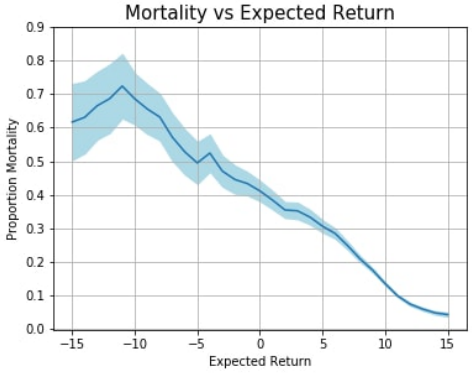

Figure 1: The relationship between expected returns — learned from observational data and actions taken by actual physicians — and the risk of mortality in the test set of 3,580 patients (see Section 5.1). The model appears to be well calibrated, with an inverse relationship between return and mortality.

6.2 Continuous State-space Models

We present the results for the two proposed networks: the Dueling Double-Deep Q-Network (Dueling DDQN) and the Sparse Autoencoder Dueling DDQN. For clarity, these are referred to as the *normal Q-N* model and *autoencode Q-N* model respectively.

6.2.1 Quantitative Value Estimate of Learned Policies

Table 2 presents the relative performance of the three policies — physician, *normal Q-N*, and *autoencode Q-N* — on expected returns and estimated mortality. As described in Section 5.2, we first obtain unbiased estimates of the value of our learned policies on the test data. The expected returns shown are $\bar{V}_{DR}^{Physician}$, $\bar{V}_{DR}^{normal\ Q\text{-}N}$, and $\bar{V}_{DR}^{autoencode\ Q\text{-}N}$. We estimate the mortality under each policy using Figure 1. As shown, the *autoencode Q-N* policy has the lowest estimated mortality and could reduce patient mortality by up to 4%. We examine a histogram of mortality counts against the first two principal components of the sparse representation (Figure 2) and observe a clear gradient of mortality counts, indicating how the autoencoder's hidden state may provide a rich representation of physiological state that leads to better policies.

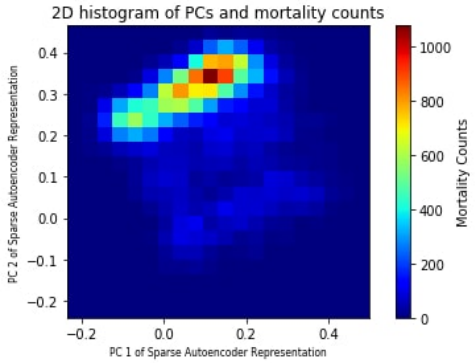

Figure 2: Histogram of mortality counts against first two principal components of sparse autoencoder representation. Note the association between these values and the eventual outcome of the patient, potentially indicating why this model was able to learn a good quality policy.

Policy	Expected Return	Estimated Mortality
Physician	9.87	$13.9 \pm 0.5\%$
Normal Q-N	10.16	$12.8 \pm 0.5\%$
Autoencode Q-N	10.73	$11.2 \pm 0.4\%$

Table 2: Comparison of expected return and estimated mortality under the physician's policy, *normal Q-N*, and *autoencode Q-N*.

6.2.2 Qualitative Examination of Learned Policies

Figure 3 demonstrates what the three policies — physician, *normal Q-N*, and *autoencode Q-N* — have learned as optimal policies. The action numbers index the different discrete actions selected at a given timestep, and the charts shown aggregate actions taken over all patient trajectories. Action 0 refers to no drugs given to the patient at that timestep, and increasing actions refer to higher drug dosages, where drug dosages are represented by quartiles.

As shown, physicians do not often prescribe vasopressors to patients (note the high density of actions corresponding to vasopressor dose = 0) and this behavior is strongly in the policy learned by the *autoencode Q-N* model. This result is sensible; even though vasopressors are commonly used

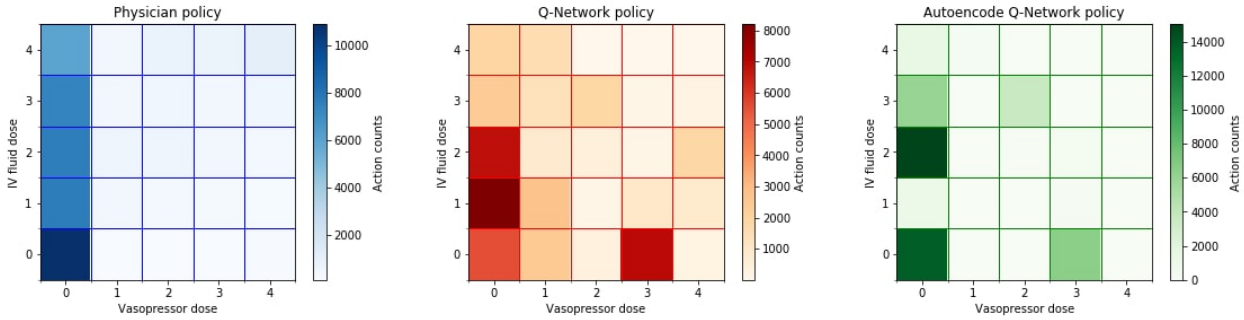

Figure 3: Policies learned by the different models, as a 2D histogram, where we aggregate all actions selected by the physician and models on the test set over all timesteps. The axes labels index the discretized action-space, where 0 represents no drug given, and 4 the maximum of that particular drug. Both models learn to prescribe vasopressors sparingly, a key feature of the physician's policy.

in the ICU to elevate mean arterial blood pressure, many patients with sepsis are not hypotensive and therefore do not need vasopressors. In addition, there have been few controlled clinical trials that have documented improved outcomes from their use (Müllner et al., 2004). The *normal Q-N* also learns a policy where vasopressors are not given in with high frequency, but that policy is less evident. There are interesting parallels between the two learned policies (*normal Q-N*, and *autoencode Q-N*). For example, both favor action (0,2) (corresponding to no IV fluids given and an intermediate dosage of vasopressor given), and action (2,3) (corresponding to a medium dosage of IV fluids and vasopressors).

6.2.3 Quantifying Optimality of Learned Policies

Figure 4 shows the correlation between 1) the observed mortality, and 2) the difference between the optimal doses suggested by the policy, and the actual doses given by clinicians. The dosage differences at individual timesteps were binned, and mortality counts were aggregated. We observe consistently low mortalities when the optimal dosage and true dosage coincide, i.e. at a difference of 0, indicating the validity of the learned policy. The observed mortality proportion then increases as the difference between the optimal dosage and the true dosage increases. Results are less reliable when the optimal dose and physician dose differ by larger amounts.

Both models appear to learn useful policies for vasopressors, with a large increase in observed mortality seen in the *autoencode Q-N* because of relatively few cases in the test set where the optimal dose and given dose differed positively by a large amount. For IV-fluids, *normal Q-N* learns a policy that shows a clear improvement over that of the physician's, indicated by the significant drop in observed mortality at the 0 mark. The *autoencode Q-N* model learns a weaker policy over IV fluids, shown by the observed mortality decreasing as the difference between dosages increases.

7. Conclusion

In this work, we explored methods of applying deep reinforcement learning (RL) to the problem of deducing optimal medical treatments for patients with sepsis. There remain many interesting areas to be investigated. The reward function in this model is quite sparse, with rewards/penalties

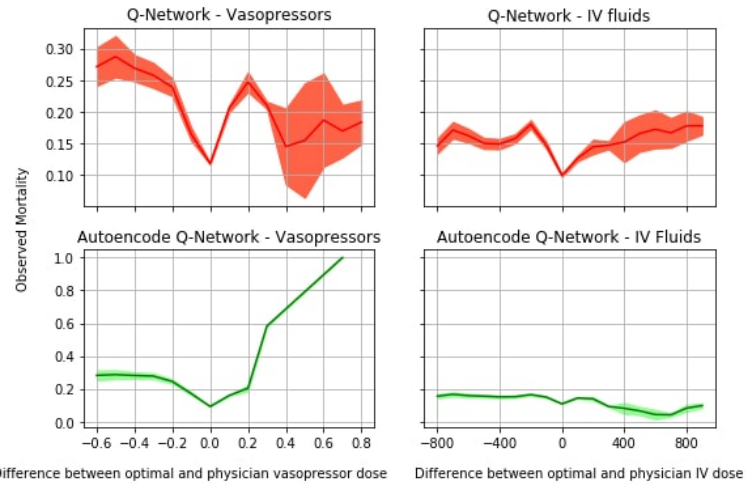

Figure 4: Comparison of how observed mortality (y-axis) varies with the difference between the dosages recommended by the optimal policy and the dosages administered by clinicians (x-axis). For every timestep, this difference was calculated and associated with whether the patient survived or died in the hospital, allowing the computation of observed mortality. In general, we see low mortality for when the difference is zero, indicating that when the physician acts according to the optimal policy we observe more patient survival.

only being issued at terminal states. To improve this, a clinically informed reward function could be used, based on patient blood counts. Another approach could be to use inverse RL techniques (Abbeel and Ng, 2010) to derive a suitable reward function based on the actions of experts (the physicians). As our dataset of patient trajectories is collected from recording the actions of many different physicians, this approach may allow us to infer a more appropriate reward function and in turn learn a better model.

Our contributions build on recent work by Komorowski et al. (2016), investigating a variety of techniques to find optimal treatment policies that improve patient outcome. We started by building a discretized state and action-space model, where the underlying states represent the physiological data averaged over four hour blocks and the action-space is over two commonly administered drugs for septic patients — IV fluids and vasopressors. Following this, we explored a fully continuous state-space/discretized action-space model, using Dueling Double-Deep Q-Networks to learn an approximation for the optimal action-value function, $Q^*(s, a)$.

We demonstrated that using continuous state-space modeling found policies that could reduce patient mortality in the hospital by 1.8–3.6%, which is an exciting direction for identifying better medication strategies for treating patients with sepsis. Our policies learned that vasopressors may not be favored as a first response to sepsis, which is sensible given that vasopressors may be harmful in some populations (D'Aragon et al., 2015). Our learned policy of intermediate fluid dosages fits well with recent clinical work finding that large fluid dosages on first ICU day are associated with increased hospital costs and risk of death (Marik et al., 2017). The learned policies are also clinically interpretable, and could be used to provide clinical decision support in the ICU. To our knowledge, this is the first extensive application of novel deep reinforcement learning techniques to medical informatics, building significantly on the findings of Nemati et al. (2016).

Acknowledgments

This research was funded in part by the Intel Science and Technology Center for Big Data, the National Library of Medicine Biomedical Informatics Research Training grant 2T15 LM007092-22, NIH National Institute of Biomedical Imaging and Bioengineering (NIBIB) grant R01-EB017205, NIH National Human Genome Research Institute (NHGRI) grant U54-HG007963, Imperial College President's PhD Scholarship, and the UK Engineering and Physical Sciences Research Council. The authors would also like to thank Finale Doshi-Velez and Luke Metz for their advice.

References

P. Abbeel and A.Y. Ng. *Inverse Reinforcement Learning*, pages 554–558. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. doi: 10.1007/978-0-387-30164-8_417. URL http://dx.doi.org/10.1007/978-0-387-30164-8_417.

Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1): 1–127, 2009. ISSN 1935-8237. doi: 10.1561/2200000006. URL http://dx.doi.org/10.1561/2200000006.

J. Cohen, J.-L. Vincent, N. K. J. Adhikari, F. R. Machado, D. C. Angus, T. Calandra, K. Jaton, S. Giulieri, J.Delaloye, S. Opal, K. Tracey, T. van der Poll, and E. Pelfrene. Sepsis: a roadmap for future research. *Lancet Infectious Diseases*, 15(5):581614, 2006.

Frederick D'Aragon, Emilie P Belley-Cote, Maureen O Meade, François Lauzier, Neill KJ Adhikari, Matthias Briel, Manoj Lalu, Salmaan Kanji, Pierre Asfar, Alexis F Turgeon, et al. Blood pressure targets for vasopressor therapy: A systematic review. *Shock*, 43(6):530–539, 2015.

A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542:115–118, 2017.

AS Fialho, LA Celi, F Cismondi, SM Vieira, SR Reti, JM Sousa, SN Finkelstein, et al. Disease-based modeling to predict fluid response in intensive care units. *Methods Inf Med*, 52(6):494–502, 2013.

Marzyeh Ghassemi, Tristan Naumann, Finale Doshi-Velez, Nicole Brimmer, Rohit Joshi, Anna Rumshisky, and Peter Szolovits. Unfolding physiological state: Mortality modelling in intensive care units. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 75–84. ACM, 2014.

Caleb W Hug and Peter Szolovits. Icu acuity: real-time models versus daily models. In *AMIA Annual Symposium Proceedings*, volume 2009, page 260. American Medical Informatics Association, 2009.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL http://arxiv.org/abs/1502.03167.

N. Jiang and L. Li. Doubly Robust Off-policy Evaluation for Reinforcement Learning. *CoRR*, abs/1511.03722, 2015. URL http://arxiv.org/abs/1511.03722.

A. E. W. Johnson, T. J. Pollard, L. Shen, L.-W. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, and R. G. Mark. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 4(160035):122, 2016.

Rohit Joshi and Peter Szolovits. Prognostic physiology: modeling patient severity in intensive care units using radial domain folding. In *AMIA Annual Symposium Proceedings*, volume 2012, page 1276. American Medical Informatics Association, 2012.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL http://arxiv.org/abs/1412.6980.

M. Komorowski, A. Gordon, L. A. Celi, and A. Faisal. A Markov Decision Process to suggest optimal treatment of severe infections in intensive care. In *Neural Information Processing Systems Workshop on Machine Learning for Health*, December 2016.

Paul E Marik, Walter T Linde-Zwirble, Edward A Bittner, Jennifer Sahatjian, and Douglas Hansell. Fluid administration in severe sepsis and septic shock, patterns and outcomes: an analysis of a large national database. *Intensive care medicine*, 43(5):625–632, 2017.

P.E. Marik. The demise of early goal-directed therapy for severe sepsis and septic shock. *Acta Anaesthesiologica Scandinavica*, 59(5):561–567, 2015.

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

Marcus Müllner, Bernhard Urbanek, Christof Havel, Heidrun Losert, Gunnar Gamper, and Harald Herkner. Vasopressors for shock. *The Cochrane Library*, 2004.

S. Nemati, M. M. Ghassemi, and G. D. Clifford. Optimal medication dosing from suboptimal clinical examples: A deep reinforcement learning approach. In *38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, August 2016.

A.Y. Ng. Sparse autoencoder, 2011. URL https://web.stanford.edu/class/cs294a/sparseAutoencoder.pdf.

N.Razavian, S.Blecker, A.M Schmidt, A.Smith-McLallen, S. Nigam, and D.Sontag. Population-level prediction of type 2 diabetes from claims data and analysis of risk factors. *Big Data*, 3(4): 277–287, 2015.

N. Prasad, L. Cheng, C. Chivers, M. Draugelis, and B. E. Engelhardt. A reinforcement learning approach to weaning of mechanical ventilation in intensive care units. 2017.

Andrew Rhodes, Laura E Evans, Waleed Alhazzani, Mitchell M Levy, Massimo Antonelli, Ricard Ferrer, Anand Kumar, Jonathan E Sevransky, Charles L Sprung, Mark E Nunnally, et al. Surviving sepsis campaign: International guidelines for management of sepsis and septic shock: 2016. *Intensive care medicine*, 43(3):304–377, 2017.

G. A. Rummery and M. Niranjan. On-line q-learning using connectionist systems. Technical report, 1994.

T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized Experience Replay. *CoRR*, abs/1511.05952, 2015. URL http://arxiv.org/abs/1511.05952.

Susan M Shortreed, Eric Laber, Daniel J Lizotte, T Scott Stroup, Joelle Pineau, and Susan A Murphy. Informing sequential clinical decision-making through reinforcement learning: an empirical study. *Machine learning*, 84(1-2):109–136, 2011.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

M. Singer, C. S. Deutschman, C. Seymour, et al. The third international consensus definitions for sepsis and septic shock (sepsis-3). *JAMA*, 315(8):801–810, 2016. doi: 10.1001/jama.2016.0287. URL `http://dx.doi.org/10.1001/jama.2016.0287`.

R.S. Sutton and A.G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998. ISBN 0262193981.

H. van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. *CoRR*, abs/1509.06461, 2015. URL `http://arxiv.org/abs/1509.06461`.

J.-L. Vincent, Y. Sakr, C. L. Sprung, V. M. Ranieri, K. Reinhart, H. Gerlach, R. Moreno, J. Carlet, J.-R. Le Gall, and D. Payen. Sepsis in European intensive care units: results of the SOAP study. *Critical Care Medicine*, 34(2):344–353, 2006.

Jason Waechter, Anand Kumar, Stephen E Lapinsky, John Marshall, Peter Dodek, Yaseen Arabi, Joseph E Parrillo, R Phillip Dellinger, Allan Garland, Cooperative Antimicrobial Therapy of Septic Shock Database Research Group, et al. Interaction between fluids and vasoactive agents on mortality in septic shock: a multicenter, observational study. *Critical care medicine*, 42(10): 2158–2168, 2014.

Z. Wang, N. de Freitas, and M. Lanctot. Dueling network architectures for deep reinforcement learning. *CoRR*, abs/1511.06581, 2015. URL `http://arxiv.org/abs/1511.06581`.

8. APPENDICES

8.1 Cohort definition

Following the latest guidelines, sepsis was defined as a suspected infection (prescription of antibiotics and sampling of bodily fluids for microbiological culture) combined with evidence of organ dysfunction, defined by a Sequential Organ Failure Assessment (SOFA) score greater or equal to 2 (Singer et al., 2016). We assumed a baseline SOFA of zero for all patients. For cohort definition, we respected the temporal criteria for diagnosis of sepsis: when the microbiological sampling occurred first, the antibiotic must have been administered within 72 hours, and when the antibiotic was given first, the microbiological sample must have been collected within 24 hours (Singer et al., 2016). The earliest event defined the onset of sepsis. We excluded patients who received no intravenous fluid, and those with missing data for 8 or more out of the 47 variables. This method yield a cohort of 17,898 patients.

8.2 Data extraction

MIMIC-III v1.4 was queried using pgAdmin 4. Raw data were extracted for all 47 features and processed in Matlab (version 2016b). Data were included from up to 24 hours preceding the diagnosis of sepsis and until 48 hours following the onset of sepsis, in order to capture the early phase of its management including initial resuscitation, which is the time period of interest. The features were converted into multidimensional time series with a time resolution of 4 hours. The outcome of interest was in-hospital mortality.

8.3 Model Features

The physiological features used in our model are presented below.

Demographics/Static
Shock Index, Elixhauser, SIRS, Gender, Re-admission, GCS - Glasgow Coma Scale, SOFA - Sequential Organ Failure Assessment, Age

Lab Values
Albumin, Arterial pH, Calcium, Glucose, Hemoglobin, Magnesium, PTT - Partial Thromboplastin Time, Potassium, SGPT - Serum Glutamic-Pyruvic Transaminase, Arterial Blood Gas, BUN - Blood Urea Nitrogen, Chloride, Bicarbonate, INR - International Normalized Ratio, Sodium, Arterial Lactate, CO2, Creatinine, Ionised Calcium, PT - Prothrombin Time, Platelets Count, SGOT - Serum Glutamic-Oxaloacetic Transaminase, Total bilirubin, White Blood Cell Count

Vital Signs
Diastolic Blood Pressure, Systolic Blood Pressure, Mean Blood Pressure, PaCO2, PaO2, FiO2, PaO/FiO2 ratio, Respiratory Rate, Temperature (Celsius), Weight (kg), Heart Rate, SpO2

Intake and Output Events
Fluid Output - 4 hourly period, Total Fluid Output, Mechanical Ventilation

8.4 Discretized State and Action-Space Model

We present here how the discretized model was built.

8.4.1 State Discretization

The data are partitioned into a training set (80%) and held-out test set (20%) by selecting a proportionate number of patient trajectories for each set. These sets were checked to ensure they provide an accurate representation of the complete dataset, in terms of distribution of outcomes and some demographic features. We apply k-means clustering to the training set, discretizing the states into 1250 clusters. As in Komorowski et al. (2016), we use a simple, sparse reward function, issuing a reward R_{max} of +15 at a timestep if a patient survives, -15 if they die, and 0 otherwise. Test set data points are discretized according to whichever training set cluster centroid they fall closest to.

8.4.2 SARSA for Physician Policy

To learn the action-value function associated with the model, we used an offline, SARSA approach with the Bellman optimality equation, randomly sampling trajectories from our training set, and using tuples of the form $< s, a, r, s', a' >$ to update the action-value function:

$$Q(s,a) \leftarrow Q(s,a) + \alpha * [r + \gamma Q(s',a') \text{ - } Q(s,a)]$$

Here, (s,a) is the current (state, action) tuple considered, (s',a') is a tuple representing the next state and action, α is the learning rate and γ the discount factor. As our state and action-spaces are both finite in this model, we represent the Q-function using a table with rows for each (s,a) tuple. This learned function was then used in model evaluation - after convergence, it represents $Q^{\pi}(s,a) = \mathbb{E}_{s' \sim T(s'|s,a)}[r + \gamma Q^{\pi}(s',a')|s_t = s, a_t = a, \pi]$, where π is the physician policy.

8.5 Continuous Model Architecture and Implementation Details

Our final network architecture had two hidden layers of size 128, using batch normalization (Ioffe and Szegedy, 2015) after each, Leaky-ReLU activation functions, a split into equally sized advantage and value streams, and a projection onto the action-space by combining these two streams together.

The activation function is mathematically described by: $f(z) = \max(z, 0.5z)$, where z is the input to a neuron. This choice of activation function is motivated by the fact that Q-values can be positive or negative, and standard ReLU, tanh, and sigmoid activations appear to lead to saturation and 'dead neurons' in the network. Appropriate feature scaling helped alleviate this problem, as did issuing rewards of ± 15 at terminal timesteps to help model stability.

We added a regularization term to the standard Q-network loss that penalized output Q-values which were outside of the allowed thresholds (± 15), in order to encourage the network to learn a more appropriate Q-function. Clipping the target network outputs to ± 15 was also found to be useful. The final loss function was:

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left(Q_{\textit{double-target}} - Q\left(s,a;\theta\right)\right)^2\right] + \lambda \cdot \max\left(\left|Q(s,a;\theta) - R_{\max}\right|, 0\right)$$

with $R_{\max}$ being the absolute value of the reward/penalty issued at a terminal timestep, and

$$Q_{\textit{double-target}} = r + \gamma Q(s', \arg\max_{a'} Q(s',a';\theta);\theta')$$

where θ are the weights used to parameterize the main network, and θ' are the weights used to parameterize the target network.

As with the discrete model, we use a train/test split of 80/20 and ensure that a proportionate number of patient outcomes are present in both sets. Batch normalization is used during training. All models were implemented in TensorFlow v1.0, with Adam being used for optimization (Kingma and Ba, 2014).

During training, we sample transitions of the form $< s, a, r, s' >$ from our training set, perform feed-forward passes on the main and target networks to evaluate the output and loss, and update the weights in the main network via backpropagation. Training was conducted for 80000 batches, with batch size 30.

8.6 Autoencoder Implementation Details

For the autoencoder, a desired sparsity ρ is chosen, and the weights of the autoencoder are adjusted to minimize $\mathcal{L}_{sparse}(\theta) = \mathcal{L}_{reconstruction}(\theta) + \beta \sum_{j=1}^{n} KL(\rho||\rho_j)$. Here, n is the total number of hidden neurons in the network, ρ_j is the actual output of neuron j, β is a hyperparameter controlling the strength of the sparsity term, $KL(\cdot||\cdot)$ is the KL divergence, and $\mathcal{L}_{reconstruction}$ is the loss for a normal autoencoder. The dimensionality of the hidden state representation was 200.