Improving Sepsis Treatment using Mixture-of-experts

Peng, Xuefeng
xpeng@g.harvard.edu

Ding, Yi
yid095@mail.harvard.edu

Wihl, David
davidwihl@g.harvard.edu

December 14, 2017

Abstract

Sepsis is main cause of mortality in the ICU and the most expensive treatment in hospitals. Patients with different physiological states respond to medical treatments very differently, and this makes sepsis management a worldwide challenge. Therefore, treatment tailored to individual patient provides significant value. In this paper, we propose a mixture-of-experts framework to further individualize sepsis treatment for each step in a patient's clinical course. The mixture model consists of kernel and DQN experts, utilizing two different gating strategies to switch actions between policies suggested from both experts: a conventional logit-based gate as well as a novel reinforcement learning based gate. Our experiments indicates that the mixture-of-experts model outperforms both kernel and DQN experts, and is thus promising for improvement in sepsis treatment. In order to reduce variance of the suggested actions, we constrain by commonly observed physician interventions. The mixed policy, including accounting for physician actions, yields a projected significant decrease in patient mortality with certain caveats.

1 Introduction

Sepsis is considered a medical emergency that requires rapid treatment. Each year, it not only costs hospitals billions of dollars but also causes tremendous patient mortalities. Individual patient respond to medical treatments very differently, and this makes sepsis management extremely challenging. Physicians have no uniform agreement on sepsis management; however, among various treatments, there are two drugs that are widely used to care for septic patients: intravenous (IV) fluid (adjusted for fluid tonicity) and vasopressor (VP). These two drugs are used to correct the hypovolemia, and counteract sepsis-induced vasodilation, respectively. Treatment decisions are critical, as they lead to extreme variation on patient mortality [WKL+14], thus, individualized treatment has a potential to improve the sepsis management.

In this work, we propose a mixture-of-experts framework aiming to tailor treatment for patients. The mixture-of-experts framework contains kernel and DQN experts using a gating strategy to switch action between policies suggested from both experts. For kernel expert, we train a recurrent auto-encoder to encode all of a patient's physiological observations during the stay in the ICU. Given a new patient, the observations are encoded to search for the nearest neighbors, which is used to derive a neighbor-based policy maximizing the survival rate. With respect to the DQN expert, we first encode patient observations into continuous state representation recurrently. Thus, the information from previous states are preserved in the current state. Then, we train the DQN using these representations. At each given state, the DQN agent takes an action with the highest Q-value, given the ultimate goal of improving the overall survival rate. We attempted two approaches to combine the policies produced from both experts. The first one, we call logistic-based switching, uses pre-trained logits to decide a probability distribution over actions from two experts, and subsequently the action with higher probability will be selected. We use WDR (weighted doubly robust) [TB16] as the unbiased off-policy estimator to evaluate the policy derived from kernel, DQN, and the mixture-of-experts. The mixture-of-experts outperforms each separate expert in the testing set. The second, which we call RL-based gating, chooses the action by searching through the state-action space to find the set of actions leading to the lowest mortality.

In particular, our contributions can be summarized as below:

- We used recurrent auto-encoder to encode patient ICU physiological states so that the interaction between these states are taken into consideration.

- We trained kernel and DQN experts to learn policies which maximize the patient survival rate.
- We created a mixture-of-experts framework that can switching actions between kernel and DQN experts to further improve the treatment for patient at any given time step.
- We created an RL-based mixture-of-experts that choses the best action between physician, kernel and DQN experts to discover the path to the lowest mortality.

2 Related Work

This work builds heavily on the prior work of Raghu, Komorowski et al. [RKC+17] in terms of using a Dueling Double-Deep Q Network (Dueling DDQN), discretized patient states and discretized physician actions for administering fluids and vasopressors. Combining kernel and reinforcement learning experts uses some of the methods applied by Parbhoo, Bogojeska, et al. [PBZ+17] in the HIV patient treatment context.

3 Data and Preprocessing

3.1 Dataset

We employ the same patient data used in [RKC+17]. The patient data is obtained from Multiparameter Intelligent Monitoring in IntensiveCare (MIMIC-III v1.4) database [JPS+16]. The dataset consists of patient from different age groups and genders with various physiological attributes. Each patient's total ICU stay is partitioned into one or more 4-hour windows, and each window counts as a set of observations. Observations describes a patient's condition (e.g. blood pressure, respiratory rate, and sequential organ failure assessment (SOFA) score) as well as the treatments (e.g. iv-fluids, vasopressor and mechanical ventilation) applied during that time window.

3.2 Data Cleaning

The dataset contains observations that are invalid; for instance, the fluids injected to a patient cannot be negative. Therefore, we eliminate patients with invalid observations. Unnecessary attributes such as chart-time, which is used to describe when the patient was moved into ICU, are excluded.

3.3 Data Preprocessing

Physiological attributes and treatment including demographics, lab-values, vital signs, and input/output events have different scales; therefore, we apply preprocessing to normalize. Table 1 describes the attributes and their corresponding preprocessing methods. After the standardization and log transformation, we rescale all values into $0 - 1$.

Table 1: Physiological attributes, treatments and corresponding preprocessing methods

Preprocessing	Attributes
Standardization	`age,Weight_kg,GCS,HR,SysBP,MeanBP,DiaBP,RR,Temp_C,FiO2_1, Potassium,Sodium,Chloride,Glucose,Magnesium,Calcium,Hb, WBC_count,Platelets_count,PTT,PT,Arterial_pH,paO2,paCO2, Arterial_BE,HCO3,Arterial_lactate,SOFA,SIRS,Shock_Index,PaO2_FiO2, cumulated_balance_tev, Elixhauser, Albumin, CO2_mEqL, Ionised_Ca`
Log transformation	`max_dose_vaso,SpO2,BUN,Creatinine,SGOT,SGPT,Total_bili, INR,input_total_tev,input_4hourly_tev,output_total,output_4hourly`

3.4 State Clustering and Treatment Discretization

We use K-means to cluster patients' physiological observations into states. Since we share the same dataset with [RKC+17], we adopt the same k-value, $k = 750$.

Furthermore, [RKC+17] shows that among all those medical interventions, there are two most critical interventions physicians usually take to treat septic patients. They are volume of intravenous (IV) fluid (adjusted for fluid tonicity) and maximum vasopressor (VP) dosage. The dosage given of those two drugs are recorded at each 4-hour window. The dosage for each of them is discretized into 5 bins, resulting in a 5×5 action space indexed from 1 to 25. Note that the first action means "no action" – neither IV-fluid nor vasopressor are prescribed.

3.5 Mortality

There are two attributes indicating mortality: `died_in_hospital` and `died_in_90_days`. Our work focuses on using `died_in_hospital` to calculate mortality rate as `died_in_90_days` introduces extra confounding factors. In the dataset, the mortality rate is 12.2% in hospital and approximately 25% after 90 days. We demonstrate the mortality rate in different age group in Figure 1. In general, higher age group associates with a higher mortality rate.

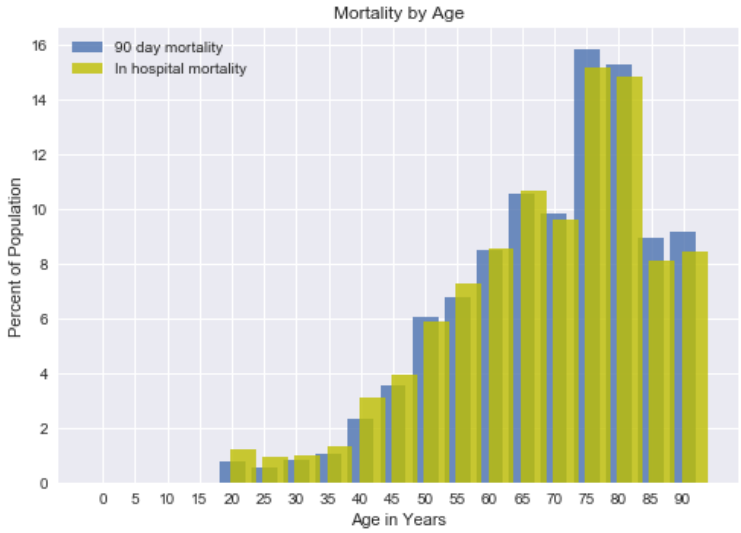
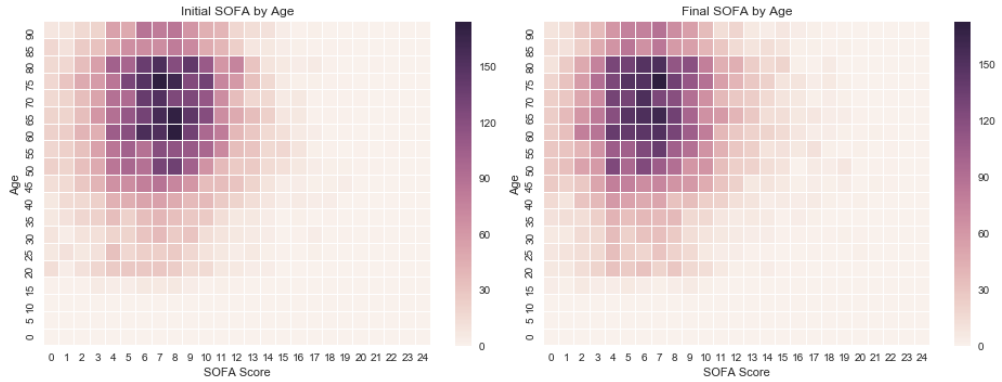


Figure 1: Mortality rate in different age groups.

3.6 SOFA Score

In order to evaluate patient stability upon entry to the ICU as well as progression of the clinical course, we use the sequential organ failure assessment (SOFA) score. The score is a combination of six different scores, one each for the respiratory, cardiovascular, hepatic, coagulation, renal and neurological systems. Each system is evaluated on a scale of $0 - 4$ where higher is worse. The per-system score is totaled. The higher total score indicates higher chance of death.

As seen in figure 2, the SOFA scores did not change dramatically between the beginning and end of most patient episodes, leading to motivation that an improved policy may lead to improved patient treatment as compared to current practices.



(a) Initial SOFA scores by age

(b) Final SOFA scores by age

Figure 2: SOFA Scores by Age

4 Methods

This section describes the methods used to learn the optimal policy for each expert and gating function. We first use a kernel method to derive group-based policy. Since [RKC+17] demonstrates the effectiveness of deep reinforcement learning approach on improving sepsis treatment, we also apply DQN, but in our case, we attempt to maximize the patient survival rate. Finally, our mixture-of-experts model is formed by combining these two methods. In the following subsections, we first describe our kernel method in details; second, we present the DQN method, and then followed by the mixture-of-experts model.

4.1 Kernel Expert

Deriving optimal policy based on a Markov Decision Process (MDP) assumes that each state transition in the patient's history is independent from all previous states, which may not be true. Therefore, we explored a direct policy learning via kernel. Our first step is transforming patient's set of observations into a single representation vector. The observations in the set are ordered by a 4-hour window since the patient arrived at ICU; thus, one patient's observations are highly temporally correlated. In order to capture this temporal information as much as possible, we applied a recurrent autoencoder. Given a new patient and one or more observations, we encode them into a single representation vector and sought the 500 nearest neighbors using Euclidean distance. Then a policy is learned directly for the given patient based on those neighbors. In summary,

- Use recurrent auto-encoder to encode each patient's all current ICU-stay histories into a representation with fixed length.
- Search 500 nearest neighbors using Euclidean distance.
- Learn policy for the given new patient that leads to the highest survival rate based on the nearest neighbors.

4.1.1 Recurrent Autoencoder

The observations collected every 4-hours from a patient in ICU stay are temporally correlated. Therefore, we use recurrent structured LSTM-autoencoder to encode those observations.

Specifically, our autoencoder utilizes one LSTM layer for both encoder and decoder. The major advantage of this structure is that each time the embedding of the previous observation and the current observation are propagated to the network together, so that the next hidden state contains the information of both current and previous observations. By doing this recurrently, the information from all previous observations can be preserved. The last hidden states from the LSTM network is our final embedding. In order to train the

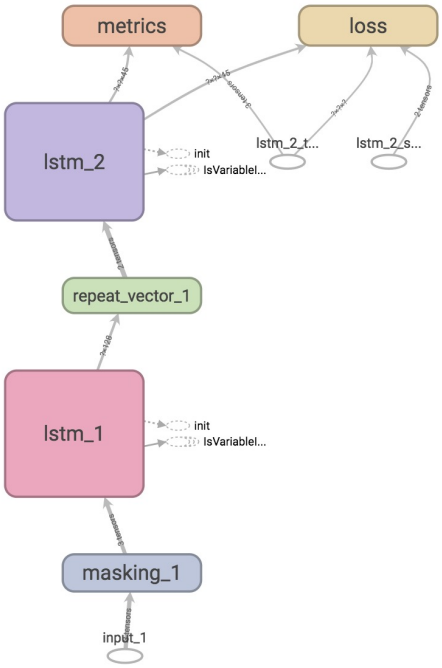
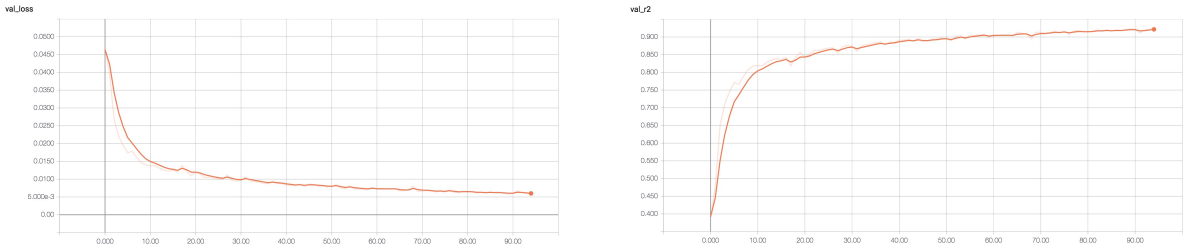


Figure 3: The LSTM-autoencoder model structure. Diagram generated from Tensorboard.

autoencoder, we then repeat the last hidden state $|observations|$ times, and decode them back; and the loss is calculated as MSE between the original observations and the decoded observations. Our final encoder and decoder both contains one LSTM layer, with 128 hidden units. Figure 3 demonstrates the model structure.

We create a feature matrix for each patient, where row i is the observation at timestep i. However, some patients have more stays than others; therefore, we find the longest stay, which is 20, and represent every patient to a $20 \times |features|$ matrix. For patients who do not have 20 stays, we pad the empty rows with 0. The padding will be masked when computing the loss.

While training the model, we apply the Adam optimizer to minimize the MSE, and use R^2 as the metric. After roughly 100 epochs, the loss on test set is decreased from 4.5×10^{-2} to 5.0×10^{-3}; and the R^2 score has increased from 0.4 to 0.92. The loss and metric are shown in Figure 4.



(a) MSE loss for each epoch

(b) R^2 score for each epoch

Figure 4: loss and metrics for training the LSTM-autoencoder

4.1.2 Neighbor-based Policy Derivation

Given arbitrary number (< 20) of ICU stay observations of a patient, we first encode them into 128-sized representation vector using the LSTM-autoencoder trained above. Then, we find 500 nearest neighbors from training dataset based on the Euclidean distance. The action that gives the highest survival rate among the same discretized state is chosen for the current state of the patient. Figure 5 illustrates this process.

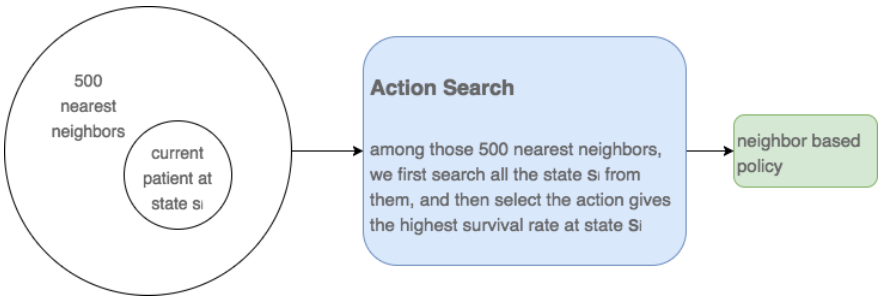


Figure 5: The process of deriving neighbor-based policy.

4.2 DQN Expert

Dueling DQN structure has been successfully applied to derive a policy that outperforms the physician policy in [RKC+17]. Therefore, we adopted the DQN structure from it. In order to differentiate the feedback of a treatment caused by a patient's current underlying physical condition, or the treatment itself, we use Dueling Double Deep-Q-Network introduced by [WSH+15]. To stabilize the training process and improve the performance, we applied three techniques: 1) reward clipping, 2) reward regularization, 3) and prioritized experience replay. In this section, we refer DQN expert as *agent*.

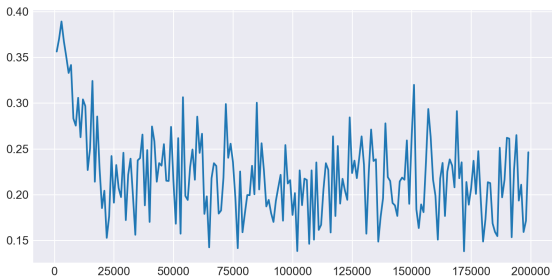
4.2.1 Continuous State Representation and Reward Formulation

To train the agent, we first treat each observation as a state so that the state space becomes continuous. [RKC+17]'s work uses a sparse autoencoder to encode the observations into a vector of length 200. However, that encoding did not consider the interaction between states. In our work, we use LSTM-autoencoder to encode each state, so that the representation of current state incorporates the information from previous states.

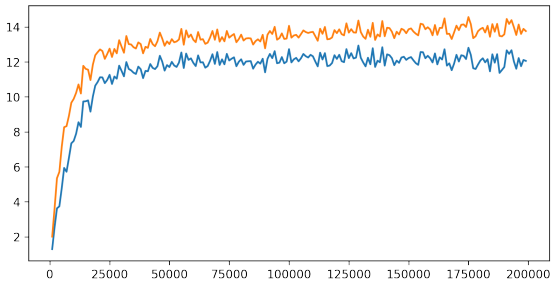
In terms of reward, for each patient, we issue 0 reward for states that are not the final state, i.e. the state where the patient is discharged from ICU. In the final state, if the patient is alive we issue a +15 reward; otherwise, we issue −15.

4.2.2 Agent Training

The agent is trained for 200,000 steps (batch size $= 32$) with the objective of minimizing the TD-error. At each given state, the agent is trained to take an action with the highest Q-value, in order to achieve the ultimate goal of improving the overall survival rate. Figure 6 demonstrates the loss and mean Q-values of physician and agent actions for each epoch. The learned policy on test set will be presented in section 5.1.



(a) The loss for each epoch.

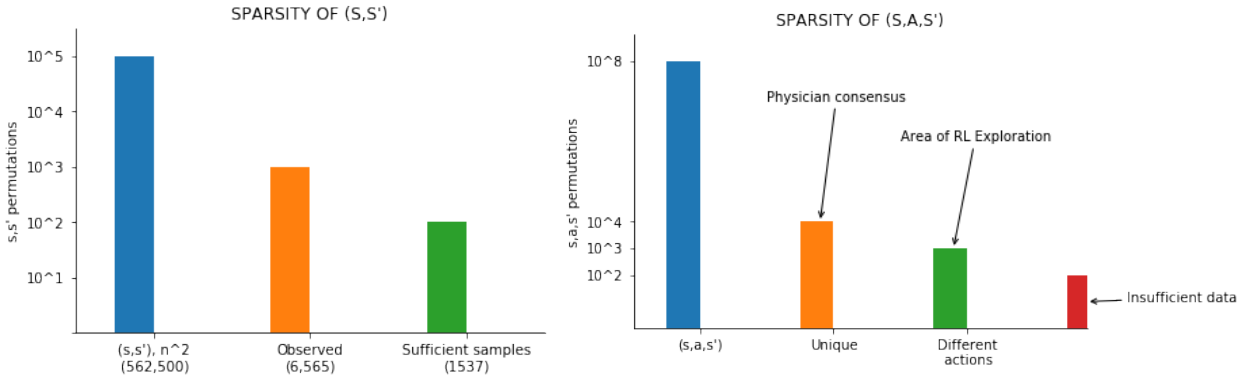


(b) Mean Q-values of physician and agent actions

Figure 6: Loss and mean Q-values of physician and agent actions for each epoch during the training phase.

4.3 Restricting Actions

Following the discretization suggested by [RKC+17], we used 750 states and 25 discrete actions. This led to a large space for all possible s, a, s' transitions ($750 \times 25 \times 750$). As shown in Figure 7, observed transitions were at least two orders of magnitude less frequent than possible transitions. Furthermore, observed transitions with sufficient samples ($n > 30$) were another order of magnitude smaller. Finally, there were certain observed transitions that did not have sufficient data in order to determine if a given physician action was effective. The area where there were sufficient examples but no physician consensus was deemed as the ideal exploration space for our mixture of experts. The percentage of physician consensus of a given action over all observed actions was a tunable hyperparameter used in our subsequent models.



(a) s, s' sparsity

(b) s, a, s' sparsity

Figure 7: Sparsity of transition space

4.4 Mixture-of-Experts

According to [PBZ+17]'s work, due to the heterogeneity of the patient dataset, it is difficult for one single model to perform well on all types of patients. Therefore, we consider combining the kernel policy as well as the DQN policy with the hope of further improving the treatment. For any given patient, our kernel expert is able to find a similar patient group, such that the derived policy works optimally for such type of patient. If the patient type is atypical, our DQN expert can effectively suggest action based on patient's continuous state representation. Thus, by taking advantage of their differing strengths, we build the mixture-of-experts model. We attempted two approaches to combine kernel and DQN experts, we call the first approach logistic-based switching. It utilizes logits to decide at any given state, taking which expert action would maximize the objective. We call the second approach RL-based gating, which applies a tree-based searching algorithm to pick actions that minimize the mortality rate.

4.4.1 Logistic-based Switching

In order to select the appropriate therapy combination, our mixture-of-experts takes the following inputs: kernel policy, DQD policy trajectory length, distance between a patient and its distance with 500th neighbor, age, gender and weight. We use WDR (weighted double robust estimator)[TB16], which is the unbiased evaluation of the learned policies, as the objective function, and update the weights using gradient ascent via Autograd [MDA15]

$$WDR(D) := \sum_{i=1}^{n} \sum_{t=0}^{\infty} \gamma^t w_i^t R_t^{H_i} - \sum_{i=1}^{n} \sum_{t=0}^{\infty} \gamma^t (w_t^i \hat{q}^{\pi_e}(S_t^{H_i}, A_t^{H_i}) - w_{t-1}^i \hat{v}^{\pi_e}(S_t^{H_i})) \tag{1}$$

Evaluation policy is computed as follows:

1. Compute probability of taking Kernel action and DQN action as $p_K = logit(h(\mathbf{x}_t^{H_i}))$, $p_D = 1 - p_K$.

2. One hot encode Kernel action and DQN action as A_K^{25} and D_K^{25};

3. Combine Kernel policy and DQN policy as $\pi(S_t^{H_i}) = p_K A_K + p_D A_D$.

In our reward formulation described in section 4.2.1, the reward signals are sparse. This sparsity may cause the instability of the WDR estimator. To stabilize the estimator, intermediate rewards, i.e. a non-zero reward issued in non-final state, are needed. Since the value function and Q function in equation 1 are meant for reducing variance, it is not necessary to be accurate. Therefore, we build MDP models based on training data and compute the V and Q functions via an approximation.

4.4.2 RL-based Gating

We also attempted a novel solution for MoE gating, namely using a reinforcement learning approach. We examined the literature but found little to no information about previous attempts to use RL as MoE gating function. We used a tree-search in this iteration. A future iteration could flatten the tree into a Q function.

In this scenario, there are effectively three experts: the set of physicians, the kernel and the DQN. Each expert suggests an action. The action with the highest reward is performed. The search continues until the best action leads to patient mortality or the maximum search depth is reached at which point the patient is presumed to survive.

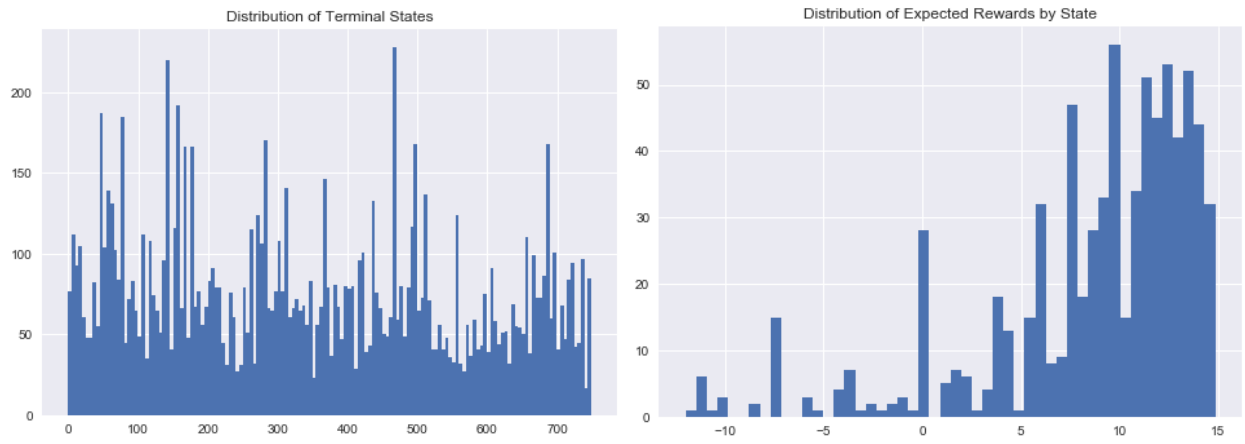
Algorithm 1 RL MoE - Tree Search Algorithm

```
procedure ChooseAction
    while depth < maxDepth do
        if physicianConsensus(state) > threshold then
            action ← physicianAction
        else
            for all experts do
                expertAction ← max(value(state))
            action ← max(expertAction)
        reward, state ← perform(action)
        if reward < 0 then
            died ← true
            break
        depth ← depth + 1
```

If there was a significant physician consensus to perform a given action in a given state, we bypassed the mixture-of-experts and selected the quorum action. The logic here is that if sufficient physicians are performing a given action, then that action is deemed to be trustworthy. The quorum threshold was a hyperparameter that afford easy tuning depending on confidence in physician choices.

This hyperparameter had a significant impact on the resulting policy. If the consensus threshold was set too low, the most common physician action was taken all the time, leaving no opportunity for the algorithmic experts to find a better path among the alternative physician choices. If the physician threshold was set too high, the physician action was rarely taken and the algorithms would attempt to find an action that incredulously led to zero mortality. After many iterations of hyperparameter tuning, we determined that physicians needed to agree on a given action 20% of the time as the best balance. Note that this value may change significantly if the semantic definition of action and state is modified.

Given the lack of reliable intermediate rewards, we attempted a very simple reward function that empirically worked well. We examined the terminal state for all episodes. The results were distributed with no apparent pattern. We then summed up the rewards found at the terminal state (± 15) and divided the by the frequency of the state occurring terminally. The result is in figure 8b which shows a useful signal. Further investigation into the merits of this approach is discussed in Section 6.3.



(a) Distribution of Terminal States

(b) Distribution of Calculated Reward by State

Figure 8: RL MoE - Reward Function

5 Policy Evaluation

We now focus on the evaluation of our policies using the test data set.

5.1 Logit Mixture-of-experts

We first show the action distributions over the test set under physician, kernel expert, DQN expert, and mixture-of-experts policies.

(a) Physician action distribution over test set

(b) DQN action distribution over test set

(c) Kernel action distribution over test set

(d) MOE action distribution over test set

Figure 9: Physician and agent action distributions over the test set

5.1.1 Action Distribution Comparisons

As shown in Figure 9, 0, i.e. no treatment action, is dominant for all policies. In the following paragraphs, we compare the action distribution from our experts and physicians respectively.

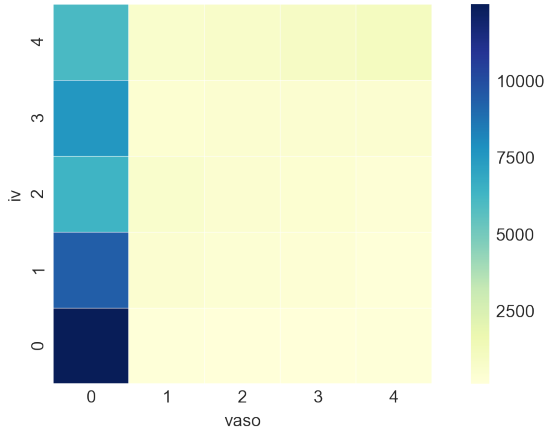
Physician and DQN actions. Physician and DQN actions. Physician tends to prescribe actions 5, 10, 15, and 20 for patients with worse conditions. In terms of the agent, 6 instead of 5 and 12 instead 10 are prescribed more frequently. Both physicians and agent are in favor of 15. For actions beyond 15, physician only prescribe 20 frequently; however, besides 20, our DQN agent also tends to give more actions in this region, such as 17, 18, and 21.

Physician and Kernel actions. Our kernel expert tends to be more conservative than the physician as it suggests no action approximately twice the physicians' frequency. Actions which are in favored by physicians, such as 5, 10, 15, 20, are also prescribed often by the kernel expert. Our kernel expert action distribution shares the general behavior of a conservative physicians.

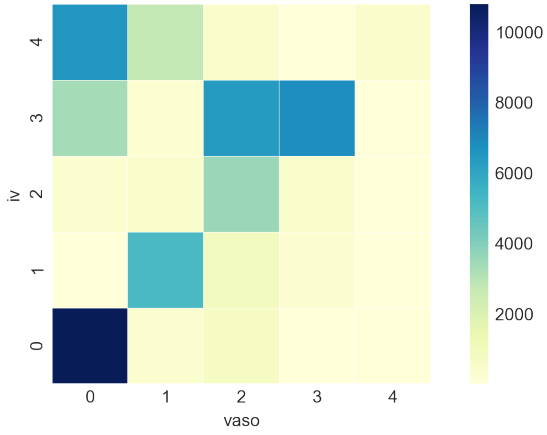
Physician and Mixture-of-Experts actions. Over the test set, the mixture-of-experts select kernel action for over 87% times; thus, the action distribution is similar to that of kernel expert. However, from 9b and 9d, we can see that MOE prescribes more frequently actions 5, 10, 15, and 20. We can see that the mixture-of-experts is functioning correctly as these actions are chosen by the DQN, not the kernel, expert.

5.1.2 IV and VP Dosage Comparisons

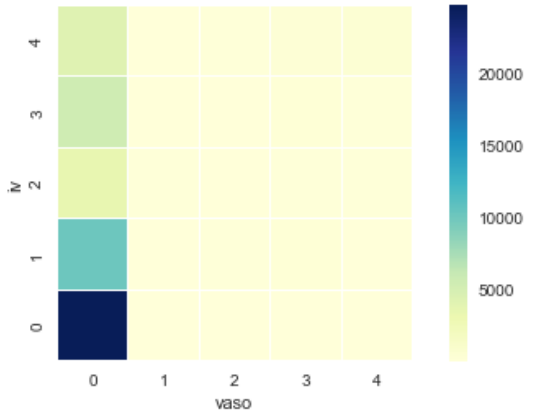
We now examine how each expert prescribes iv-fluids and vasopressors. Figure 10a shows that iv-fluids are largely used in the physician policy, and the usage as well as the dose volume of vasopressor is increased while higher dose of iv-fluids are given.



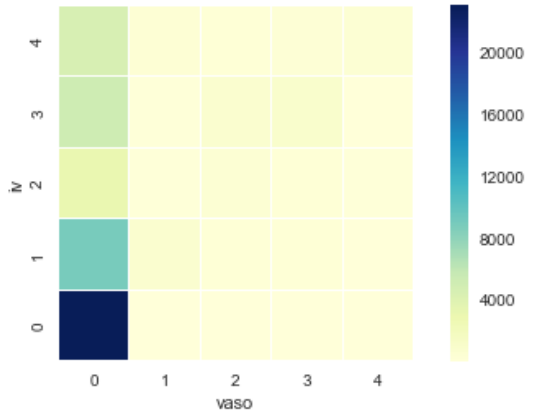
(a) Physician action heatmap over test set



(b) DQN action heatmap over test set



(c) Kernel action heatmap over test set



(d) MOE action heatmap over test set

Figure 10: Physician and agent action heatmap over the test set

With respect to the DQN policy, this tendency becomes more obvious. Note that in Figure 10b, the

actions represented by these diagonal cells are used more often than in physician policy. However, just like the physician, the agent is scrupulous about prescribing the maximum dose of vasopressor. In general, the agent is suggesting that the iv-fluids and vasopressor should be prescribed comparatively more frequently. Kernel policy is close to the physician policy except it is even more conservative in giving high dosages for both IV-fluids and vasopressor. Finally, since the kernel actions are chosen 7 times more often than DQN actions, the mixture-of-experts acts much akin to the kernel expert. However, we do can observe that in 10d, the diagonal cells appear deeper, implying that the MoE chooses the action suggested by the DQN experts that for patient under certain physiological states where higher dosages of vasopressor is required.

5.1.3 Off-policy Evaluation using WDR

We run WDR on policies suggested from physician, kernel, DQN, and MOE. Table 2 shows that over both training and testing sets, mixture-of-experts policy achieves the highest unbiased estimation.

Table 2: WDR estimate for different policies

Policy	Train	Test
Physician	6.89	7.13
DQN	8.09	8.65
Kernel	9.35	4.59
Mixture of Expert	11.16	8.81

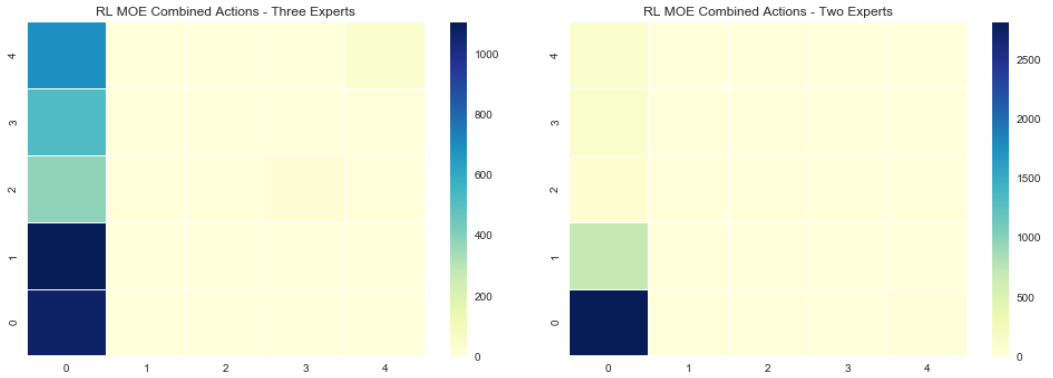
5.1.4 Logistic Switching Weight Interpretation

To enable the mixture-of-experts to switch actions, we need to assign a probability distribution over actions from experts based on a given patient's features. For each patient, the features we use are trajectory length (i.e. number of observations), furthest distance from its 500 nearest neighbors, age, gender, and weight.

After training, the weights for features are: trajectory length= 1.67, max distance to neighbor = -1.40, age=0.74, gender=2.18, weight=1.87. As trajectory length increases, more patient information is available, which assists in finding reliable neighbor policies. Therefore, for longer trajectories, the logit prefers the Kernel action over the DQN action. In contrast, when the distance to the furthest neighbor is large, the Kernel policy will be unreliable, so the logit prefers DQN actions.

5.2 RL Mixture-of-experts

The resulting three expert mix provides a reasonable balance between a physician-only and an algorithmic-only policy, especially when configured appropriately by the consensus hyperparameter. Using only two algorithmic experts (kernel, DQN), the resulting policy is more conservative than the logit-based gating function, favoring the kernel more frequently than the DQN.



(a) RL MoE - Three Experts (b) RL MoE - Two Experts (Kernel, DQN)

Figure 11: Resultant Policy from RL-based MoE

Table 3: RL MoE Results - Three Experts

Physician Consensus	Est. Mortality	Consensus Occurred
10%	13.4%	100.0%
20%	10.0%	23.1%

Examining the frequency of consensus between the different experts was enlightening as the algorithmic approaches agreed more frequently than the physician chosen action. Investigating this discrepancy is a potential area of further research.

Table 4: Action Agreement Frequency - Three Experts

	Kernel	DQN
Physician	4.1%	3.3%
Kernel		45.4%

By using the MoE (including accounting for physician policy), we project a significant improvement in patient mortality (13.4% vs. 10.0%). This appears very promising so caveats are discussed in the sections below.

6 Limitations

6.1 Use of WDR as Logit Gradient Function

As the authors of WDR acknowledge in a subsequent paper[LTB], WDR is prone to instability:

> Recent work such as doubly robust and MAGIC, shows that we can get benefit from combining importance sampling method with model value. However they all assume that they have only one model. In case we have several different models, which is common in some complex domains, it may be hard to select the best one from them, and may lose the potential benefit from all models. we present a evidence example to show that select model by simply minimizing the notation of error in previous estimator (MAGIC) can fall into a wrong model, which suggest that selecting a best model for off-policy policy evaluation is non-trivial and worth of further exploration.

WDR furthermore requires an intermediate reward function, which was not readily available in the dataset. We iterated through numerous rounds of feature engineering in attempting to find an intermediate reward that had reasonable predictive value, ultimately choosing change in SOFA score and change in Arterial Lactate over a single period. (See Appendix B for a discussion of our intermediate reward investigation).

6.2 Intermediate Rewards

Our MoE approach, using both conventional gradient gating as well as an RL-based approach were stymied by the lack of high quality value function or intermediate rewards. As we have seen numerous times in other research, RL success hinges on having a stable and predictive value function. Once the value function is reliable, the policy iteration becomes significantly easier. Without a reliable value function, policy iteration is impossible. This is analogous to neural net-based supervised learning: without a quality training set, a supervised learning algorithm cannot perform well. Intuitively, a better intermediate reward function and a better loss function would yield even more promising results from a mixture-of-experts.

6.3 RL MoE Reward Function

While the results of the RL-based MoE appear encouraging, and the resulting policy seems reasonable, it is recommended to perform a more thorough evaluation of the predictability of the expected reward metric. An ROC analysis and a Brier Calibration test would give greater credibility to this reward function and the RL-based MoE in general.

6.4 RL MoE has Markov and Deterministic Assumptions

The RL MoE makes two assumptions: that the state transitions are Markov, which may not be the case. We also assume that actions are deterministic, where a given action will always lead to a given new state. A future version would relax this assumption by modeling stochasticity based on the known distribution of s, a, s' from the training set. Modeling stochasticity would also be a more realistic means of measuring mortality.

7 Discussion

As shown in figure 10, each expert has different behaviors. The kernel method is most conservative, tempering variance of different physician actions. The DQN has higher variance which leads to more aggressive actions for patients that may need more intervention earlier. The overall combination yields a generally conservative approach. The heatmaps occlude the cases where DQN has found a better treatment course for patients that are outliers. Given the small number of patients in this category, the DQN is an appropriate means of finding appropriate pathways that are outside the capabilities of the conservative kernel method. Per our original objective, treatment has become more personalized.

Perhaps in combination with another team using Inverse Reinforcement Learning, it would be possible to find an appropriate reward function to drive both the WDR and the RL based gating functions.

Even with the very simplistic reward function, the experimentation with RL-based gating yielded several interesting ideas worth pursuing. The specific questions raised include:

- Could the human specialist (physician in this case) be considered simply another expert in the mixture? What priority should physician actions be given over algorithmic suggestions? Examining where disagreements occur between human and between algorithms may yield interesting insight into the decision making process for both approaches.
- Assuming a viable reward function exists, is it possible to simply evaluate all observed physician actions and choose the policy based on which patients are receiving the reward (ie. lower mortality in this case) more frequently? In other words, especially in the case of medicine, what kind of policy choices can be offered to the physician at the moment of decision making that provide useful, reliable information regarding interventions made by peers in similar circumstances?

8 Future Work

Creating a robust simulation environment is highly desirable in order to ascertain the effectiveness of novel treatment strategies. While there has been some efforts in using Recurrent Conditional GANs for generation

of sepsis training data [EHR17], a model-based approach is possibly less prone to the instability often found in GANs. If a physiological model is difficult due to lack of domain knowledge, an alternative data-driven approach may be the use of microsimulation, which has been previously used in sepsis and ICU treatment evaluation although not in an RL context [CKM+04, SKS+07].

The use of restricted actions based on observed physician actions would be interesting to apply to other treatment domains, even as a tool to evaluate consistency of treatment. This could also be used as a means of determining more precisely which interventions led to lower mortality. Finally, it may serve a useful as a clinical tool for physicians to consult before performing interventions, answering questions such as: "What did other physicians do for patients in this state? With what frequency? What was the mortality?"

The idea of using an RL-based MoE gating function, especially in problems that lend themselves to an RL approach (as opposed to a differentiable loss function) is worth investigating. One recent paper [SMM+17] from Google Brain (Hinton, Dean, et al.) alludes to this but offers no specifics as to a means of policy iteration.

The area of off-policy policy evaluation merits further investigation, especially in cases where only replay of past history is available. A robust mechanism would open RL to a much larger set of problem spaces while potentially offering higher safety by minimizing risky explorations. The high variance of current approaches leave room for significant improvement.

As mentioned previously, a more rigorous evaluation of the RL-based reward function would be desirable to ensure the credibility of this approach.

9 Conclusion

Sepsis management is an interesting and engaging problem space, with potential for significant human impact. We ambitiously attempted several approaches, some of which were novel. We feel there are interesting areas meriting further exploration, both from a sepsis treatment standpoint as well from a general machine learning approach. Initial results are promising showing a significant decrease in projected patient mortality. Combining human and algorithmic policies in domains with high consequences such as healthcare can lead to both better, more specific policies that are potentially safer and more effective than current practices.

Acknowledgements

We would like to express our sincere gratitude to Prof. Doshi-Velez, Dr. Matthieu Komorowski and Omer Gottesman for their advice, guidance and candid questioning of the approaches we attempted. We especially appreciate their patience as this project changed direction and emphasis. We would also like to thank the other students in Harvard CS282R Fall 2017 for their insights, encouragement and feedback.

Appendix A Simulation Environment for RL

Given the sample inefficiency of training a DDQN, we initially evaluated creating a simulator of a patient's clinical course. This would have allowed experimentation with novel treatments for which there was no empirical data. Simulation has been used successfully in many RL scenarios, most visibly in the use of the Atari simulation environment [BNVB12] for the creation of DQNs [MKS+13] / DDQNs [WSH+15].

Unfortunately, the model-based Sepsis simulators available today (for example [BRS+04]) are used primarily for physician training and are not appropriate for use as an RL training environment.

We did attempt to create synthetic data by mimicking the observation distributions found in the source data set. However, we abandoned this effort because we felt that it would not be providing a robust representation of novel interventions since there was no supporting data and may suffer from biases. The analysis of the state distributions did prove useful for restricting actions.

Appendix B Intermediate Reward Investigation

In order to determine an intermediate reward function, we explored several options.

B.1 Observations Correlated with Mortality

Performing a correlation yielded the following most correlated observations:

Table 5: Observations Most Closely Correlated with 90 Day Mortality

Observation	Correlation
SOFA	0.25
BUN	0.23
age	0.21
Elixhauser	0.16
SIRS	0.15

B.2 Classification Results

We then attempted a variety of classifiers using the above features and measured the F1 score. We also evaluated accuracy against a larger set of features. The accuracy of multiple classifiers did not sufficiently exceed the baseline. The following table summarizes the results for the best F1 score and the relevant hyperparameter.

Table 6: Classification Performance on Training Set

Classifier	Hyperparameter	F1 Score	Accuracy
Baseline	N/A	N/A	0.77
Logisitic Regression	$C = 0.1$	0.07	0.79
Random Forest	Trees = 100	0.33	0.78
kNN	$k = 10$	0.29	0.77
SVM	$C = 0.0001$	0.07	0.77
XGradBoost	estimators= 200	0.30	0.79

Given the low correlation, low F1 score, and baseline accuracy, we did not see strong predictive value for mortality in the observations.

References

[BNVB12] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *CoRR*, abs/1207.4708, 2012.

[BRS+04] G Breuer, R Riss, T Schröder, S Mönk, and J Schüttler. The intensive care simulator: a new teaching-concept to train severe sepsis management. *Deutsche medizinische Wochenschrift (1946)*, 129(48):2586–2589, 2004.

[CKM+04] Gilles Clermont, Vladimir Kaplan, Rui Moreno, Jean-Louis Vincent, Walter T Linde-Zwirble, Ben Van Hout, and Derek C Angus. Dynamic microsimulation to model multiple outcomes in cohorts of critically ill patients. *Intensive care medicine*, 30(12):2237–2244, 2004.

[EHR17] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[JPS+16] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3, 2016.

[LTB] Yao Liu, Philip S Thomas, and Emma Brunskill. Model selection for off-policy policy evaluation.

[MDA15] Dougal Maclaurin, David Duvenaud, and Ryan P Adams. Autograd: Effortless gradients in numpy. In *ICML 2015 AutoML Workshop*, 2015.

[MKS+13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[PBZ+17] Sonali Parbhoo, Jasmina Bogojeska, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez. Combining kernel and model based learning for hiv therapy selection. *AMIA Summits on Translational Science Proceedings*, 2017:239, 2017.

[RKC+17] Aniruddh Raghu, Matthieu Komorowski, Leo Anthony Celi, Peter Szolovits, and Marzyeh Ghassemi. Continuous state-space models for optimal sepsis treatment - a deep reinforcement learning approach. *CoRR*, abs/1705.08422, 2017.

[SKS+07] Görkem Saka, Jennifer E Kreke, Andrew J Schaefer, Chung-Chou H Chang, Mark S Roberts, Derek C Angus, GenIMS Investigators, et al. Use of dynamic microsimulation to predict disease progression in patients with pneumonia-related sepsis. *Critical Care*, 11(3):R65, 2007.

[SMM+17] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[TB16] Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148, 2016.

[WKL+14] 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.

[WSH+15] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*, 2015.