Predicting Optimal Sedation Control With Reinforcement Learning

by

Anuhya Vajapey

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2019

Author . . . .
Department of Electrical Engineering and Computer Science
May 24, 2019

Certified by . . . .
Peter Szolovits
Professor
Thesis Supervisor

Accepted by . . . .
Katrina LaCurts
Master of Engineering Thesis Committee

Predicting Optimal Sedation Control With Reinforcement Learning

by

Anuhya Vajapey

Submitted to the Department of Electrical Engineering and Computer Science
on May 24, 2019, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Administering sedation to patients to avoid underdosing and overdosing is an important clinical task that remains hard to control due to lack of precision in current methods of measuring sedation. The type of drugs administered, the procedure the patient is undergoing, patient characteristics (age, gender, weight, height), even genotypes can affect the way the patient's body processes the sedation administered. Currently, sedation is administered by an attending anesthesiologist who sets a target sedation level and continuously monitors the patient with an EEG and adjusts the target level accordingly. In this thesis, I apply Fitted Q-Iteration to learn a Reinforcement Learning Model that takes in a patient's current state and predicts the dosage of sedation to administer at each second during the procedure to keep the patient's physiological variables within clinically normal ranges. I experiment with different state and action representations to demonstrate how different choices affect the policy learned by the Reinforcement Learning Model. I evaluate the results qualitatively and quantitatively through the implementation of Doubly Robust Policy Evaluation.

Thesis Supervisor: Peter Szolovits
Title: Professor

Acknowledgments

I would like to thank my advisor Professor Peter Szolovits. I'm extremely grateful to have such a kind and supportive advisor such as Pete. His patience with students and continuous encouragement made this experience incredibly enlightening and enjoyable. I'm incredibly grateful for the collaborative environment he fosters in his lab and the independence he gives his students to be explore their research interests.

I would like to say a huge thank you to my direct mentor and labmate Matthew McDermott. He guided me through every step of the way from debugging code to understanding the nitty gritty of the theory and even reviewing my thesis. The countless brainstorm sessions we had, constant communication and continuous support he gave, and the incredible patience with which he answered all my questions, made my MEng experience incredibly enriching and transformative. He helped me grow both academically and personally and I am extremely grateful.

I would like to thank Dr. Pedro Gambus, the clinical collaborator on this research. He was incredibly helpful in understanding the clinical context of this task. The data analyzed was collected by his team in Barcelona.

I would also like to thank Willie, Emily, Tristan, and everyone in MEDG. Every chance I had to work with them was a pleasure and they welcomed me into this community of medical research.

Lastly, I would like to thank my family for their support and always encouraging me to persevere and go for my goals undaunted.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Diagnostic procedures, such as gastrointestinal endoscopy often require sedation to ensure that patients remain free of stress and pain and remain still during the procedure. Sedation is often administered through IV by an attending anesthesiologist or nurse anesthesiologist through a Target-Controlled Infusion (TCI) pump, which uses mechanistic models to attempt to maintain a target sedative concentration in patients [4].

Sedation consists of a combination of powerful anesthetic and analgesic drugs, such as the common combination of propofol and remifentanil. Anesthesia causes a lack of response in a patient to harmful stimulus (such as undergoing a surgery) and analgesia results in relief from pain [16]. Too little sedation can cause post traumatic stress disorder and complications during the procedure but too much sedation can slow down recovery and cause adverse postoperative outcomes [11]. However, it is difficult to determine the optimal depth of sedation since drug absorption varies from person to person and the combination of drugs given have synergistic effects that are hard to model [13].

To address this issue, many systems have been developed to assess patients' level of sedation. For anesthetic agents which mainly alter brain function, electroencephalographic signal (EEG) derived measures are used. Recently, more complicated models that take in more inputs have been developed to monitor sedation since it is hard to quantify pain relief and model the synergistic effects of different drug combinations

[17].

Sedation was administered to prevent gag reflex during the insertion of the endoscopy tube for patients undergoing gastrointenstinal endoscopy. In this thesis work, I implement a Reinforcement Learning model that learns an anesthetic/analgesic administration policy for patients undergoing gastrointestinal endoscopy. The RL model's goal is to maintain the patient's state within a clinician defined set of normal ranges for certain hemodynamic variables. Furthermore, the model should not underdose and result in a patient having gag reflex at the insertion of the endoscopy tube. I evaluate the results through the U-curve method and Doubly Robust Policy Evaluation but off-policy evaluation is a challenging domain that requires more careful analysis.

1.1 Thesis Overview

Chapter 2 provides background on sedation control models, reinforcement learning, and policy evaluation. Chapter 3 details the dataset analyzed and explains the cleaning and preprocessing measures taken. Chapter 4 discusses the various models implemented and the results. Chapter 5 outlines the evaluation models and techniques employed to examine the results. Chapter 6 concludes my thesis work and guides the future direction of this approach.

Chapter 2

Background

There is a huge variability in sedation response across patients as a result of many different factors: age, gender, ethnicity, disease, surgical intervention, combination of drugs administered, etc. There is no exact measure to determine how deeply a patient is sedated, and there is a delay in response to the anesthesia that makes the task of giving the optimal dose and assessing it even more difficult [26]. The following measures have been developed to use as indicators for estimating the degree of sedation.

Bispectral Index (BIS) of the EEG is one common way of measuring the hypnotic effects of the drugs. BIS is a dimensionless number scaled from 0 (electric silence) to 100 (fully awake) [1].

Auditory Evoked Potential (AEP) index is a single numerical parameter derived from the AEP, electrical potentials evoked in the auditory pathway in response to sound stimuli, in real time that can also serve to monitor anesthetic depth.

Ramsay Sedation Score (RSS) is a scale of 0 to 6 that is often used to assess sedation by checking a patient's responsiveness to a stimulus. The meaning of different RSS scores is shown in Table 1.1 below. The target score at the end of a procedure is usually either 3 or 4.

Score	Level of sedation
6	Dangerously agitated and uncooperative.
5	Agitated.
4	Restless but cooperative
3	Calm and cooperative
2	Responsive to touch or name
1	Responsive only to noxious stimuli
0	Unresponsive

Table 2.1: RSS scores and corresponding patient response

Nociceptive (gag) response to the introduction of endoscopy tube. Gag response is an automatic reflex triggered by the nervous system to protect the body from damaging stimuli.

2.1 Sedation Control Models

Current standards for assessing sedation often involve measuring patient's response and behavior on a scale like RSS or Motor Activity Assessment Scale, which are subjective to the judgment of the anesthetist [7]. More computational pharmokinetic pharmodynamic (PK-PD) models have been developed to model the effects of sedation. These models often used EEG-derived measures (BIS monitoring and M-Entropy) to augment the traditional behavior scale [19, 5]. BIS was used in a proportional-differential control algorithm that titrated propofol to achieve target BIS=50. The closed-loop algorithm, which measured the error between current BIS value and the target and adjusted the propofol target accordingly, maintained the target BIS with smaller adjustments than the clinicians [14]. Another model used M-Entropy analysis on EEG signals to achieve a similar goal of maintaining patient level of sedation within specified ranges for more periods of time than manual adjustments by the clinicians [15]. Auditory evoked potentials have also been used in differential-control models in various studies [10].

EEG monitoring is not always used for determining depth of sedation. Instead, one study calibrated a linear SVM model on electrocardiogram data to Richmond

Agitation Sedation Scale scores to determine sedation levels [19]. Another study used a computerized test, CogState, to assess a patient's psychomotor function, attention, visual memory, and working memory to understand how propofol and remifentanil are processed by a patient [2]. An adaptive neuro fuzzy inference system was implemented by Gambus et al. to analyze the relationship between a variety of sedation measures (AAI, BIS, RSS, Index of Consciousness) and the predicted effect-site concentrations of propofol and remifentanil [4].

A major limitation of these methods is that it is still hard to predict how the various measures will change with a dose of anesthesia since each patient has different responses to the sedation administered based on many different factors. One study tried to actually model the changes in BIS after anesthesia was administered to a patient by applying machine learning with patient information and drug concentrations targeted as features and successfully predicted BIS very closely to the actual BIS value [3]. Several studies have attempted to study how drugs are processed in different sets of patients and with different combinations of drugs. Hannam et al. found that a specific genotype thought to affect sedation response and noxious stimulation was not a significant factor in determining the level of respiratory depression caused by propofol and remifentanil administration, but age was [8]. One study applied an LSTM model to understand the discrepancy between the predicted effect-site concentration and measure BIS during intravenous anesthesia when propofol and remifentanil were administered and showed that a deep learning model was more accurate at predicting BIS than the traditional pharmodynamic models used [12]. Another study showed that the type of anesthesia administered (IV vs general) and different surgical stimuli affect the hemodynamic control a patient has upon waking [25]. In sum, there are many factors that affect a patient's response to anesthesia (genetics, age, surgical intervention, type of anesthesia, combination of anesthetics administered, etc.) and many models have been proposed to solve this issue but sedation control still remains a process of trial and error.

2.2 Reinforcement Learning

Reinforcement Learning is a set of algorithms to determine optimal policies over a Markov Decision Process (MDP) with states, actions, and rewards. MDPs model an environment as a set of states ($\mathcal{S}$) and actions ($\mathcal{A}$) to control the system's state in a sequential decision making problem to maximize total cumulative rewards collected ($\mathcal{R}$). MDPs make a Markov Assumption that the future dynamics of a system are dependent only on its present state, not the past history of states. In a Reinforcement Learning approach, we can define states to be over a set of discrete timesteps for a total length of time T.

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, ..., S_t] \tag{2.1}$$

The goal of an RL process is to maximize the total cumulative rewards achieved. In order to do this we use a value function that predicts the value of a state $V_t(s)$. A value function is necessary because the value of state must denote the expected long-term return, not just the reward at that state. Formally, we control this through discounting future rewards, specified by $\gamma : [0, 1]$. We can write a function to denote the rewards accumulated between time t and T as follows:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=t+1}^{T} \gamma^{k-(t+1)} R_k \tag{2.2}$$

The γ denotes the discount factor on future rewards. In economics, this corresponds to uncertainty about the future or how much to rely on future estimates being accurate during the calculation of return. Computationally, γ may help aid in convergence of the RL algorithm by ensuring that massive end state rewards don't swamp immediate rewards during training. A policy, π, is the mapping of states to probabilities of selecting each action. The value of a state under a policy π is described by the state-value function v_π.

$$v_\pi(s) = \mathbb{E}_\pi \left[G_t | S_t = s \right] = \mathbb{E} \left[\sum_{k=t+1}^{T} \gamma^{k-t-1} R_k \middle| S_t = s \right], \forall s \in S \tag{2.3}$$

The action-value function calculates the expected reward of following a policy starting from a state s and taking an action a. It maps (state, action) pairs to returns and is described by q_π.

$$q_\pi(s,a) = \mathbb{E}_\pi\left[G_t|S_t = s, A_t = a\right] = \mathbb{E}\left[\sum_{k=t+1}^{T} \gamma^{k-t-1} R_k \middle| S_t = s, A_t = a\right] \tag{2.4}$$

Let $p(s', r|s, a)$ be the transition probability of moving to s' from state s by taking action a and getting reward r. We can write the *Bellman Equation* for v_π as follows:

$$v_\pi(s) = \mathbb{E}_\pi\left[G_t|S_t = s\right] = \sum_a \pi(a|s) \sum_{s',r} p\left(s', r|s, a\right)\left[r + \gamma v_\pi(s')\right], \forall s \in S \tag{2.5}$$

The equation simply computes the probability of the (a, s', r) triple occurring, weights the return expected by the probability, and sums over all the possibilities to get the expected value of the policy. A similar equation can be written for the action-value function. Solving an RL task means that we want to find a policy that achieves the greatest return. According to the Bellman equation, we can see that the value of a state under an optimal policy must equal the expected return from the best action from that state. Formally we can write the solution to the Bellman equation as follows:

$$\begin{aligned} v_*(s) &= \max_{a \in A(s)} q_{\pi *}(s,a) \\ &= \max_a \mathbb{E}_\pi\left[G_t|S_t = s, A_t = a\right] \\ &= \max_a \sum_{s',r} p\left(s', r|s, a\right)\left[r + \gamma v_\pi(s')\right] \end{aligned} \tag{2.6}$$

The Bellman equation for the q-function is as follows:

$$\begin{aligned} q_*(s,a) &= \mathbb{E}_\pi[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a')|S_t = s, A_t = a] \\ &= \sum_{s',r} p(s', r|s, a)[r + \gamma q_*(s', a')] \end{aligned} \tag{2.7}$$

When the transition dynamics between states are known (p), an exhaustive search can be conducted using Dynamic Programming to solve the Bellman optimality equation

v_* or q_*.

Off-policy Learning

However, in the clinical setting, we dont know if the actions the clinician took were the best possible actions in relation to our goals and more importantly, the state space is not discrete. This means we cannot just use Dynamic Programming to try all possible state/action combinations and solve the problem. We cannot just learn the policy the clinician followed and need to do "off-policy" learning. Furthermore, since we do not know the transition probability distribution or reward function, we use a "model-free" approach that does not explicitly compute the transition probability from state-action pairs to the next state. Q-learning is a well known model-free off-policy algorithm that updates the Q-function (or Value function) through trial and error by keeping a table of Q-values and updating function based on the returns it actually sees [27]. The function is initialized arbitrarily by the programmer on the first iteration and gets better through many iterations. The update rule is formalized below:

$$Q(s_t, a_t) \leftarrow (1 - \alpha) * Q(s_t, a_t) + \alpha(r_t + \gamma * \max_a Q(s_{t+1}, a)) \tag{2.8}$$

Here α is the learning rate and dictates how much to update Q-function with the new Q-function vs keeping the old Q-function. Essentially the agent observes the current state, selects an action a, observers the subsequent state, collects the reward, and adjusts the Q_{n-1} values using the learning rate. With a sufficiently large number of iterations, the Q-function has been shown to converge [27]. The algorithm I implemented in this thesis is a variation of Q-learning where the Q-function is approximated and updated according to Eqn. 2.8. Fitted Q-Iteration(FQI) is a batch-mode reinforcement learning algorithm where the entire data is available at the start, allowing for the use of any supervised learning regression algorithm to approximate the Q-function. The algorithm uses one-step transition tuples

$$\mathcal{F} = \{(\langle s_t^n, a_t^n, s_{t+1}^n \rangle, r_{t+1}^n), n = 1, ..., |\mathcal{F}|\}$$

to learn a sequence of function approximators $\hat{Q}_1, \hat{Q}_2, ..., \hat{Q}_K$ where the optimal policy after K iterations will be: $\operatorname*{argmax}_{a \in \mathcal{A}} \hat{Q}_K(s, a)$.

Algorithm 1: Fitted Q-Iteration

```
Input  : $\mathcal{F} = \{s_t^n, a_t^n, s_{t+1}^n, r_{t+1}^n\}, n = 1, ..., |\mathcal{F}|$
         Regression parameters $\theta$
Output: $\theta$
Intialize $Q_0(s_t, a_t) = \mathcal{R}$;
$\mathcal{S} = \langle (s_t, a_t), Q_0 \rangle$, $\forall s_t \in \mathcal{F}, a_t \in \mathcal{A}$;
for $k \leftarrow 1$ to $K$ do
    $f$ = regress($\langle \mathcal{S}, Q_k \rangle, \theta$);
    for $i \leftarrow 1$ to $n$ do
        $Q_k(s_t^i, a_t^i) \leftarrow r_{t+1}^i + \gamma \max_{a' \in \mathcal{A}}(f(\langle s_{t+1}^i, a' \rangle, \theta))$;
        $S \leftarrow \langle (s_t, a_t), Q(s_t, a_t) \rangle$;
    end
end
```

The state representation, action space, and reward function greatly dictate the success of the algorithm in effectively modeling real world transition dynamics and learning an optimal policy. For this reason, I experiment with different state representations, different action spaces, and different reward functions. Another important factor is the regressor used to learn the Q-value function and can also be varied with a simple linear regression, decision trees, neural networks, etc.

2.2.1 Evaluation

In a clinical setting, evaluation of a learned policy (hereby referred to as "evaluation policy") is difficult because we cannot run the policy and test it to see how it works. Off-policy evaluation is usually done in one of two ways. The first way is to fit an MDP model from the data via regression and evaluate the policy against the fitted model. After fitting an MDP model, we could estimate the value of the target policy through recursively solving the Bellman Equations. However, learning an MDP is often really hard since many state action pairs predicted may never be observed in the data and applying a function approximation to solve this problem will introduce inherent bias. Thus, in the case of clinical data, we apply the second class of approaches, Importance

Sampling, which solves this problem by providing an unbiased estimate of the target policy's value by taking an average estimate of the trajectories [9].

Importance Sampling

The goal of Importance Sampling (IS) is to learn the value of the evaluation policy based on data trajectories given by the behavior policy (clinician policy). Importance Sampling reweights evaluation policy's returns to account for differences in the likelihood of the returns between evaluation and behavior policies. The approach takes a probability of the first t steps of history H occurring under evaluation policy divided by the probability of them occurring under the behavior policy.

$$\begin{aligned} V_{IS} &= \rho_{1:H} \cdot \left(\sum_{t=1}^{H} \gamma^{t-1} r_t \right) \\ V_{step-IS} &= \sum_{t=1}^{H} \gamma^{t-1} \rho_{1:t} r^t \end{aligned} \tag{2.9}$$

where $\rho_t := \frac{\pi_e(a_t|s_t)}{\pi_b(a_t|s_t)}$ is the importance ratio and 1:H is used to denote stepping over each timestep in a patient's history. Eqn. 2.9 formalizes the general IS estimator and the step-wise IS estimator. If the evaluation policy and behavior policy greatly differ, the IS estimator will have a high variance and have a large range of importance weights. Doubly Robust is a variation of Importance Sampling that reduces the variance by providing good estimates if either the model is accurate or the behavior policies are known (hence, "doubly" robust). The Doubly Robust estimator (DR) takes an estimated reward function and importance weight and calculates the value as follows:

$$V_{DR} = \hat{V}(s) + \rho \left(r - \hat{R}(s,a) \right) \tag{2.10}$$

where $\rho := \frac{\pi_e(a_t|s_t)}{\pi_b(a_t|s_t)}$ and $\hat{V}(s) := \sum_a \pi_e(a|s) \hat{R}(s,a)$. If $\hat{R}$ is a good estimate of r, the magnitude of $r - \hat{R}(s,a)$ will be much smaller than that of r, resulting in lower variance than IS. The step-wise DR estimator is shown in Eqn. 2.11 where $\hat{Q}(s_t, a_t)$

is given or calculated via regression from training data set similar to $\hat{R}$.

$$V_{DR}^{H+1-t} = \hat{V}(s_t, a_t) + \rho_t \left(r_t + \gamma V_{DR}^{H-t} - \hat{Q}(s_t, a_t) \right) \tag{2.11}$$

The algorithm I implemented for Doubly Robust Policy Evaluation is as follows:

Algorithm 2: Doubly Robust Policy Evaluation

```
Input  : from training data: $\hat{Q}$, $\hat{R}$, $\pi_e$, $\pi_b$
Output: $V_{DR}$
initialize: $V_{DR} = 0$, $\gamma = .8$
for (s,a,r) $\in H$ do
    $\hat{V} = 0$;
    for $a \in \mathcal{A}$ do
        $\hat{V} += \pi_e(a|s) * \hat{R}(s,a)$;
    end
    $V_{DR} += \hat{V} + \frac{\pi_e(a)}{\pi_b(a)} * (r + \gamma V_{DR} - \hat{Q}(s,a))$;
end
```

U-curve

A more qualitative evaluation method is the U-curve method which associates the difference between the behavior policy and evaluation policy to some outcome. The data in this study is from patients who received sedation (propofol and remifentanil) prior to undergoing gastrointestinal endoscopy. An outcome in this case would be whether the patients had a gag response when the tube is inserted. A plot of the differences in outcome based on plot of difference between behavior policy's recommended actions and clinician policy's recommended actions is shown to determine if the RL model was better at identifying underdosed patients.

2.2.2 Prior Work in RL

Reinforcement Learning methods have been used in a variety of studies for different clinical tasks. Prasad et al. used Reinforcement Learning to predict the weaning of mechanical ventilation for patients in the ICU [22]. They use an off-policy reinforcement learning algorithm with fitted Q-iteration to determine the best possible action from sedation drug and dosage, ventilator settings, initiation of a spontaneous breathing trial, or extubation at each patient state. Nemati et al. developed a deep reinforcement learning algorithm that models the internal belief of a patient's state with a partially observable Markov decision process to learn an optimal heparin dosing policy from EMR data [20]. Raghu et al. used a deep reinforcement learning approach to deduce optimal treatment policies for patients with sepsis in the ICU by binning actions into 5 bins for vassopressors and IV fluids and using 4hr windows for the state representation [23]. Moore et al. tested Q-learning to control anesthetic dosing in a volunteer study as an alternative to the proportional-integral-derivative controllers. The model smoothed a patient's past BIS values over 15s windows and used a discretized action space of propofol targets to choose from to achieve a target BIS [18]. Padmanaban et al. conducted a similar study on 30 simulated patients where Q-learning was applied to create an RL-based controller to administer IV sedatives that also took into account the synergistic effects of the combination of drugs used for sedation [21]. Another interesting application of RL is actually Inverse RL which essentially looks at the observational data and formulates the RL model setup to be used to learn a policy. Yu et al. apply FQI with a Gradient Decision Tree regressor to learn ventilator weaning policy from ICU data and then apply inverse RL to learn the reward functions and compare the policy learned to the clinicians' policy [28].

There are a lot of studies that applied RL in clinical settings with many different MDP formulations. However, none of the studies have applied RL to predict actions at such a short temporal resolution as I do in this work. Furthermore, the reward function I use consists of not just the sedation measures (RSS, BIS) but also important hemodynamic variables such as partial pressure of Carbon Dioxide (PCO2), Oxygen Saturation (SpO2), and Heart Rate (HR). In practice, these variables are constantly monitored to make sure a patient doesn't experience hemodynamic depression for an extended period of time to prevent complications and serve as useful calibration measures for sedation dosing.

Chapter 3

Data Overview

3.1 Background

The dataset consists of an anonymized database of 380 patients undergoing deep sedation (Propofol and Remifentanil) for gastrointestinal endoscopy in Hospital Clinic of Barcelona, Spain. Propofol and remifentanil were administered targeting the effect site using a TCI system. Data collection started five minutes before starting drug administration.

Table A.1 describes the full form of the variable acronyms used. Hemodynamic variables (continuous heart rate, non-continuous arterial blood pressure), raw and processed EEG (BIS, AAI derived features), pulse oximetry derived SpO2, and PCO2, were all recorded. RSS was also assessed by a clinician at irregularly chosen times throughout the procedure. In addition, continuous infusion rate and total infusion volume was tracked for both Propofol and Remifentanil, along with PKPD determined plasma and effect site concentrations of each drug (CpPROPO/CpREMI and CePROPO/CeREMI, respectively). Whether or not the patient gagged in response to introduction of endoscopy tube (this was considered a nociceptive response) was also recorded for all subjects. A subset of 210 patients were also monitored with transcutaneous pCO2 (Sentec) and also had a screening for A118G polymorphism in OPRM gene altering μ receptor spatial conformation and conditioning resistance to the effects of opioids. Age, weight, height, and gender were also collected.

Rugloop was the software used to download the data in real time with a resolution of 1 datum/second except for ABP, 1 measurement every three minutes. RSS mesurements, gag response, tube introduction and any other clinically relevant events were entered as a comment with a predefined clinical code to allow analysis. Data from all monitors and events were stored offline for further synchronization and analysis.

3.2 Exploratory Analysis

Since we are concerned with administering sedation, we will be using InfRatePROPO and InfRateREMI, the drug infusion rates set on the TCI machine, as the variables we care to track in our action space. A big problem with this is that the rates recorded are often extremely close to 0. Some preliminary analysis on the measurements taken across all the patients showed that 80.34% are 0s, after removing 7.34% of the data which were NaNs. Furthermore, 90.62% of the data is $\leq$1. After aggregating the Propofol and Remifentanil infusion rates across all patients, I plotted histograms of the rates to visualize any skew in Fig. 3-1. The first row shows histograms of the drugs without any removal of data and has a huge skew towards 0. The second row shows the same rates after we drop the bottom 5% of the data. More of the underlying data begins to show once this step is done. The third row shows the data after removal of the lowest 5% and removal of rates equal to 0. This distribution is more conducive to fitting an RL model. Without any processing, the action that the model will predict will almost certainly always be 0. Furthermore, we don't lose any information by removing these datapoints because they occur as an artifact of the data collection process which begins 5 minutes prior to start of the surgery (before anesthesia was administered) and ends some time after the surgery is completed.

The reward function will take as an input the patient's hemodynamic variables recorded throughout the surgery. Fig. 3-2 shows a few of the main clinical variables that are collected throughout a patient's stay in the ICU and the variables doctors are constantly monitoring to make sure they are within the clinically normal ranges. The left axis shows the rate and the right axis shows the volume of the drug accumulated.

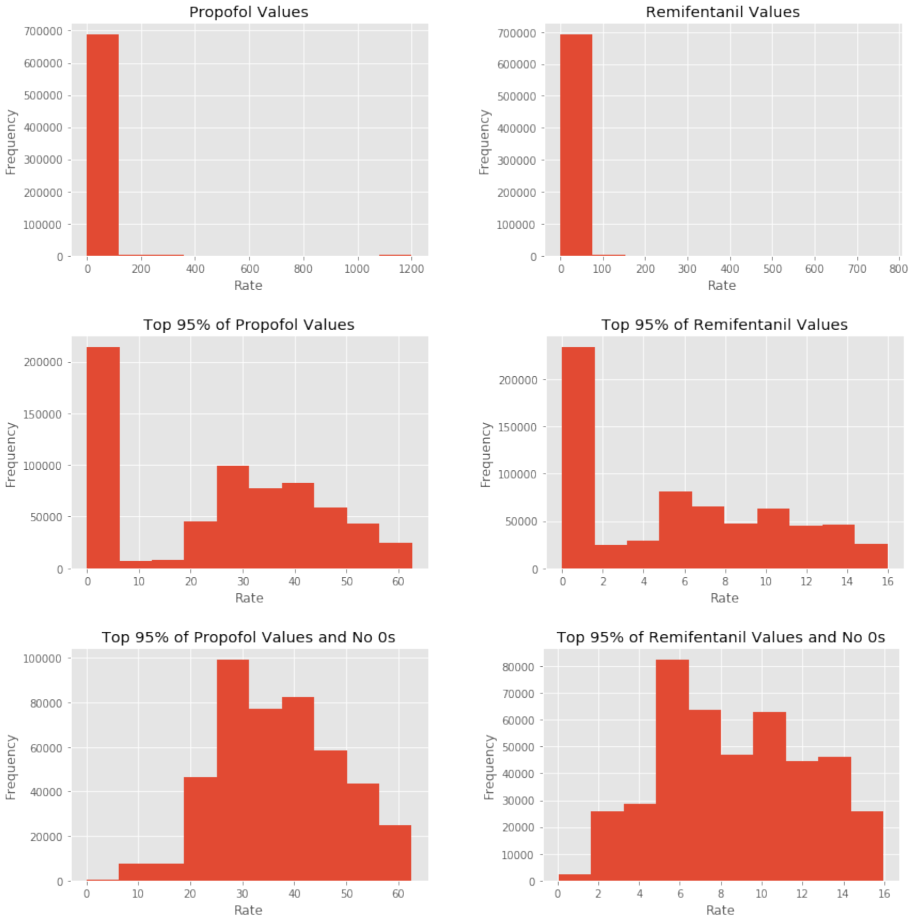


Figure 3-1: Histograms of Propofol and Remifentanil Values

The shaded yellow regions correspond to "decompensatory" regions where the values are not clinically normal. Note that the drug rates are continuously increased even when the hemodynamic variables seem to be abnormal. This could indicate overdosing and something we want the RL model to reduce.

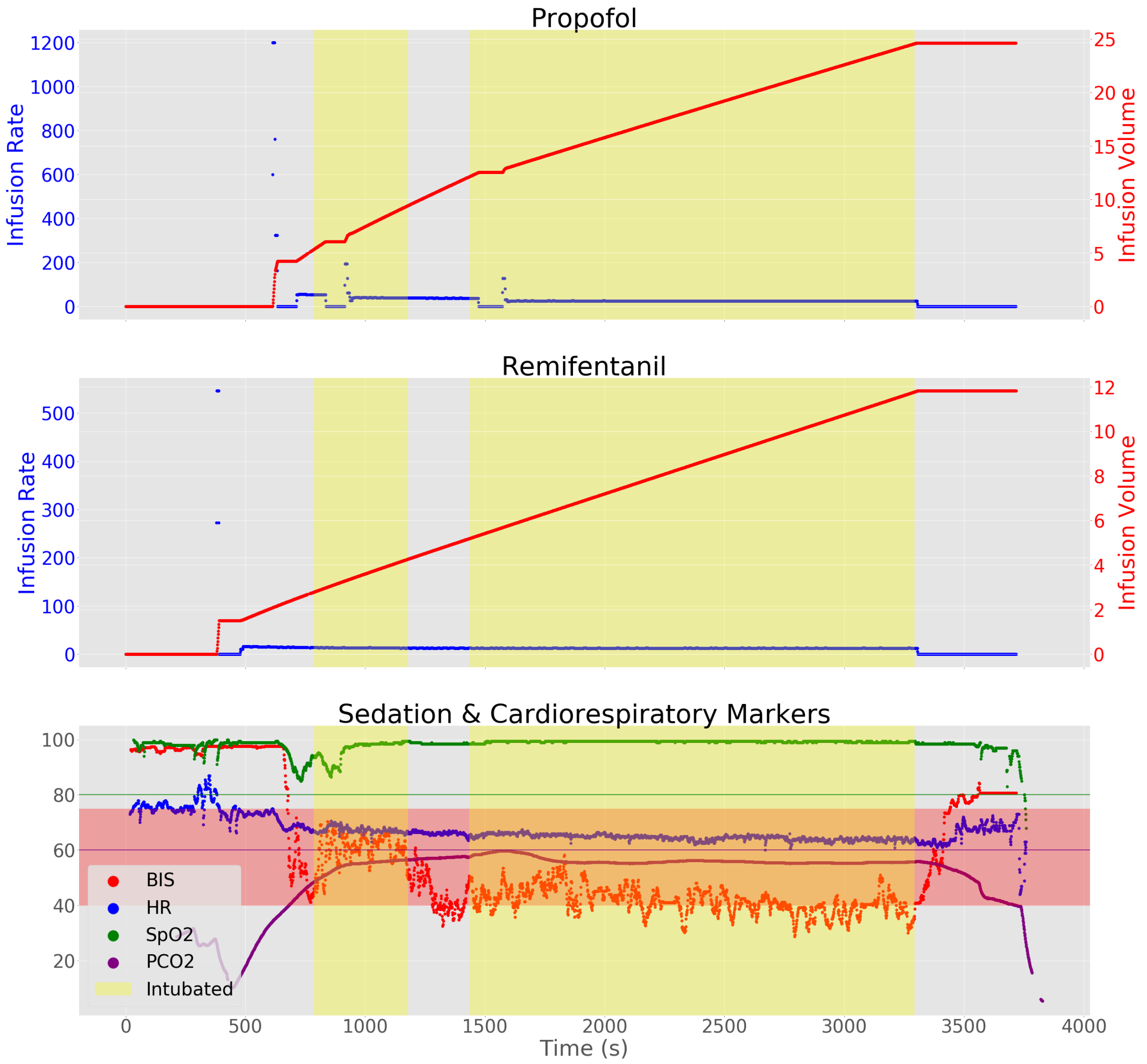


Figure 3-2: Plot of a patient's hemodynamic variables across the length of stay and the corresponding plots of propofol and remifentanil administered. The highlighted regions correspond to the decompensatory regions where the variables are out of the normal ranges.

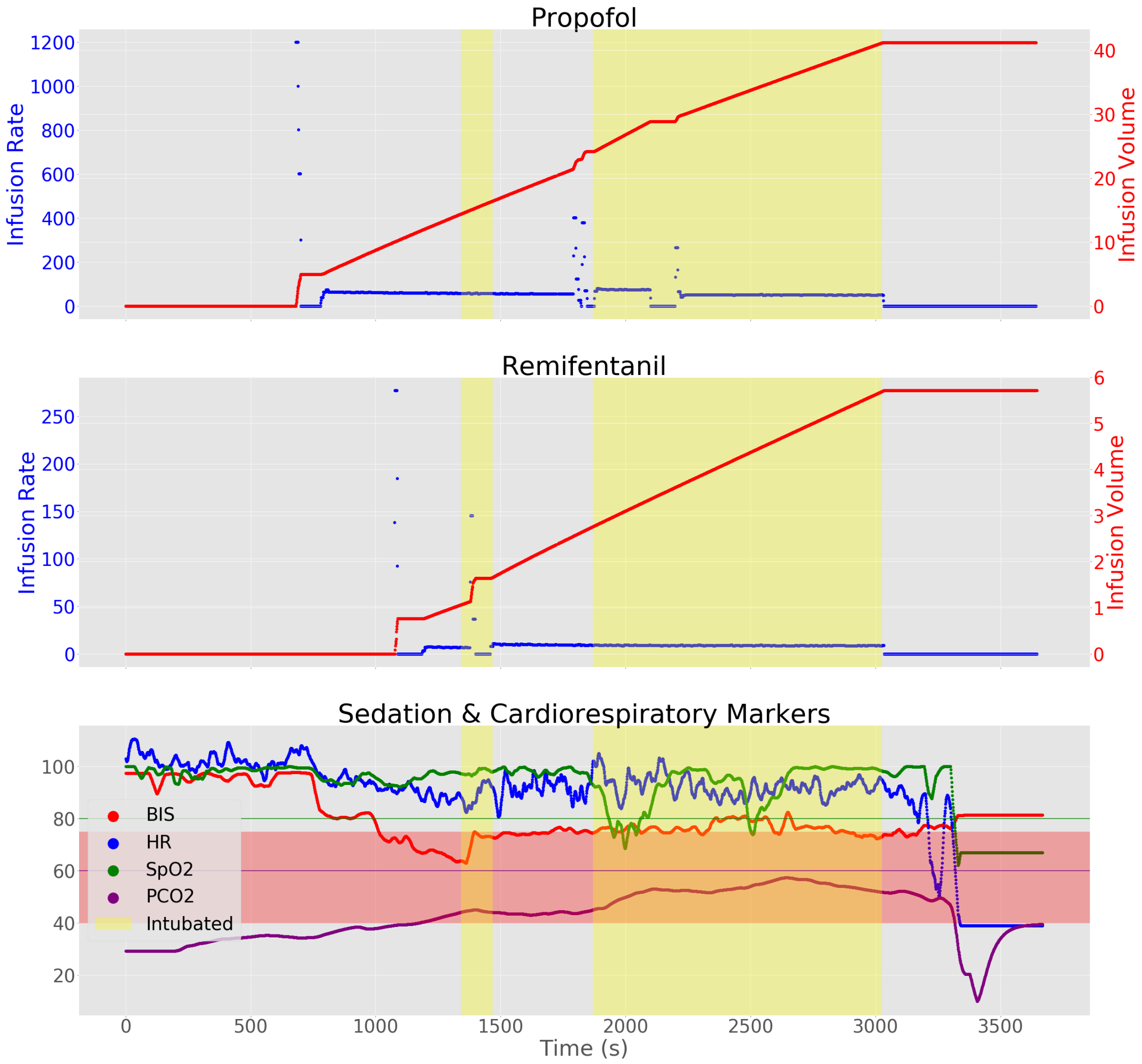


Figure 3-3: Plot of a different patient's hemodynamic variables across the length of stay and the corresponding plots of propofol and remifentanil administered. The data has been smoothed by taking a moving average over 30s window.

3.3 Cleaning

Various features were collected at different frequencies, which means there will be missing data in different time frequencies. There are also long regions of missingness where sensors fall off during the procedure and aren't immediately put back on. For any data that was collected at a resolution of 5 seconds or less, linear interpolation was applied to get data at a resolution of 1 second. Afterwards, any missing data was imputed via forward fill imputation within each patient's data. Each patients collected data was also trimmed at the start of data collection and end of data collection to remove 0s. A window of 30s of recorded measures for either Propofol or Remifentanil was needed before the patients data was considered since each patient often had a few minutes of 0 values recorded, 15s of max rates recorded, and 0s recorded again until the procedure started. This is simply an artifact of the controller and as such I trimmed each patient's data till at least 30 seconds of continuous non-zero measurements were recorded.

Chapter 4

Experiments and Results

The following sections describe the various models implemented. There were many variations on the state space, action space, and reward function that I tried and each combination greatly impacted the resultant policy.

For the following three models, I first discretized the drug rates into actions to fit the MDP. As described in section 3.2, the rates recorded were greatly skewed towards 0. Furthermore, the rates increase continuously. To discretize, I decided to bucket the action ranges into a fixed number of bins, which can be done in two approaches. One approach is to make the bins the same sizes and let the volume of rates that fall in each bin vary. The second approach is to use quantiles which can vary in size of each bin but have equal volume of rates fall in each bin. Because the data is skewed, binning into equal sizes, not volume, resulted in an arbitrary output action bin of 0 for both propofol and remifentanil because the 0 bins had the majority of the values for each drug. After experimenting with different number of quantiles, I found that 7 was the maximum number of bins that resulted in an equal volume of rates across all the bins. Thus, for each of the drugs, I first determined the ranges for the 7 quantiles and binned the rates to achieve a tuple as defined by (propofol $\in$ {0,1,2,3,4,5,6}, remifentanil $\in$ {0,1,2,3,4,5,6}). Then I applied a one-hot encoding of the resulting 7x7 action space to create a one-hot 49x1 vector that denoted the action taken at a timestep. Bin number, ranges for each bin, and volume of rates are shown in Table 4.1.

Table 4.1: Bins, ranges, and frequency used to discretize $\mathcal{A}$

Propofol		
Bin	**Range**	**Volume**
0	[0, 21.438)	75083
1	[21.438, 28.077)	75099
2	[28/077, 32.849)	75086
3	[32.849,38.825)	75059
4	[38.835, 44.652)	75121
5	[44.652, 53.948)	75085
6	[53.948, 1200.0]	75105

Remifentanil		
Bin	**Range**	**Volume**
0	[0, 2.73)	75088
1	[2.73, 5.628)	75027
2	[5.628, 6.86)	75122
3	[6.86, 9.156)	75122
4	[9.156, 11.193)	75074
5	[11.193, 13.716)	75092
6	[13.716, 767.76]	75113

While final bucket seems to have a large range, the distribution of rates that fell in each bucket are also skewed as can be seen in Fig. 3-1. It is clear that in bin 6, even though the range is wide, the values are mostly centered around the lower bound except for a few outliers falling at the upper bound.

Other setups I tried but are not described in further detail include running a model with actions as a set of {0,1,2,...,48} where each number encoded a combination of 7x7 binned propofol and remifentanil dosing. The result from this model was to always dose bin 1 for propofol and 0 for remifentanil. I also tried an action space with 10 equally sized bins, which resulted in the model always outputting (0,0) for all timesteps. The action space that seemed to work the best was using 7 quantiles for each drug and I used this $\mathcal{A}$ in the 3 models described below, which vary in the state representation and slightly in the reward function.

Figure 4-1: Distributions Across Bins for Propofol

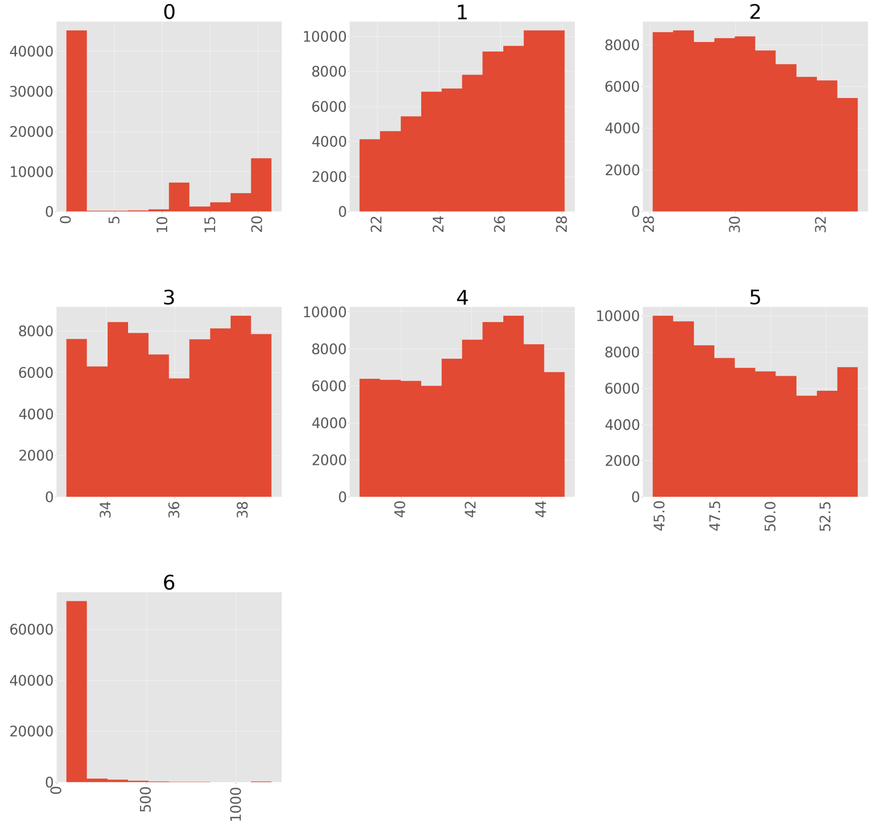
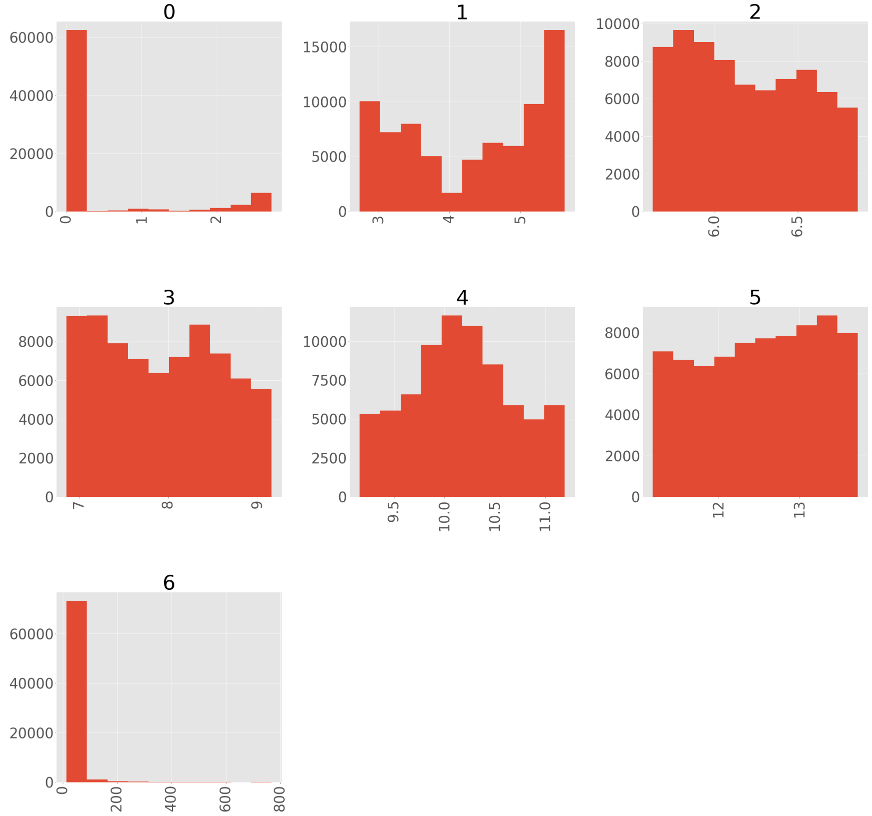


Figure 4-2: Distributions Across Bins for Remifentanil



4.1 Single Step Model

The state space was a single timestep consisting of all the variables listed below. For each step, the current state was represented by a 29x1 vector. $\mathcal{S}$: [‘AAI’, ‘Age’, ‘BIS’, ‘BSA’, ‘BSAAI’, ‘BSBIS’, ‘CePROPO’, ‘CeREMI’, ‘CpPROPO’, ‘CpREMI’, ‘EMGAAI’, ‘EMGBIS’, ‘GABRB3’, ‘GAG’, ‘Gender’, ‘Height’,‘LBM’, ‘NIBPdia’, ‘NIBPmean’, ‘NIBPsys’, ‘OPRM1’, ‘PCO2’, ‘RSS’, ‘RespiRate’, ‘SQI09’, ‘TUBE’, ‘Weight’, ‘SpO2’, ‘HR’].

The action space was the one-hot encoded quantiles as described earlier. It was a 49x1 vector that denoted the combination. The reward function I used in this model is a squared mean error penalty. It checks whether, at the next state, BIS is between 40 and 75, RSS between 3.5 and 4.5, PCO2 is at most 60, SpO2102 is at least 80, and Heart Rate is between 45 and 100.

```
def calc_reward(row):
    rew = 0
    rew +=(row["BIS"]-57)**2
    rew +=(row["RSS"]-4)**2
    if row["PCO2"]>60:
        rew+=(row["PCO2"]-60)**2
    if row["SpO2"]<80:
        rew+=(80-row["SpO2"])**2
    rew+=(row["HR"]-72)**2
    return -rew
```

I tried a LinearRegression and DecisionTreeRegressor from sklearn as the q-function approximators. I ran the LinearRegressor for 100 and 200 iterations and the DecisionTreeRegressor for 200 iterations. Below is a plot of the medians of the binned actions for propofol and remifenatil that the RL model suggested vs doctor did, Q-values and the difference in Q-values between doctor’s policy and RL policy, and a plot of the respiratory markers for one patient. The medians for each bin were determined from the training data. Figure 4-3 depicts the actions predicted for one patient at every timestep for the duration of the procedure the patient underwent. The RL model seemed to always underdose both propofol and remifentanil when compared to the actions taken by the clinician. The Q-values vary at the start but converge toward the end to a higher Q-value than the clinician policy's Q-value. Figure 4-4 shows the same data but smoothed over a 30s moving window to visualize a less noisy depiction of the policies for this patient. It is interesting to note that the RL model starts off with dosing smaller amounts of drugs at the start and then increases the dosage compared to the clinician who always doses the same amount.

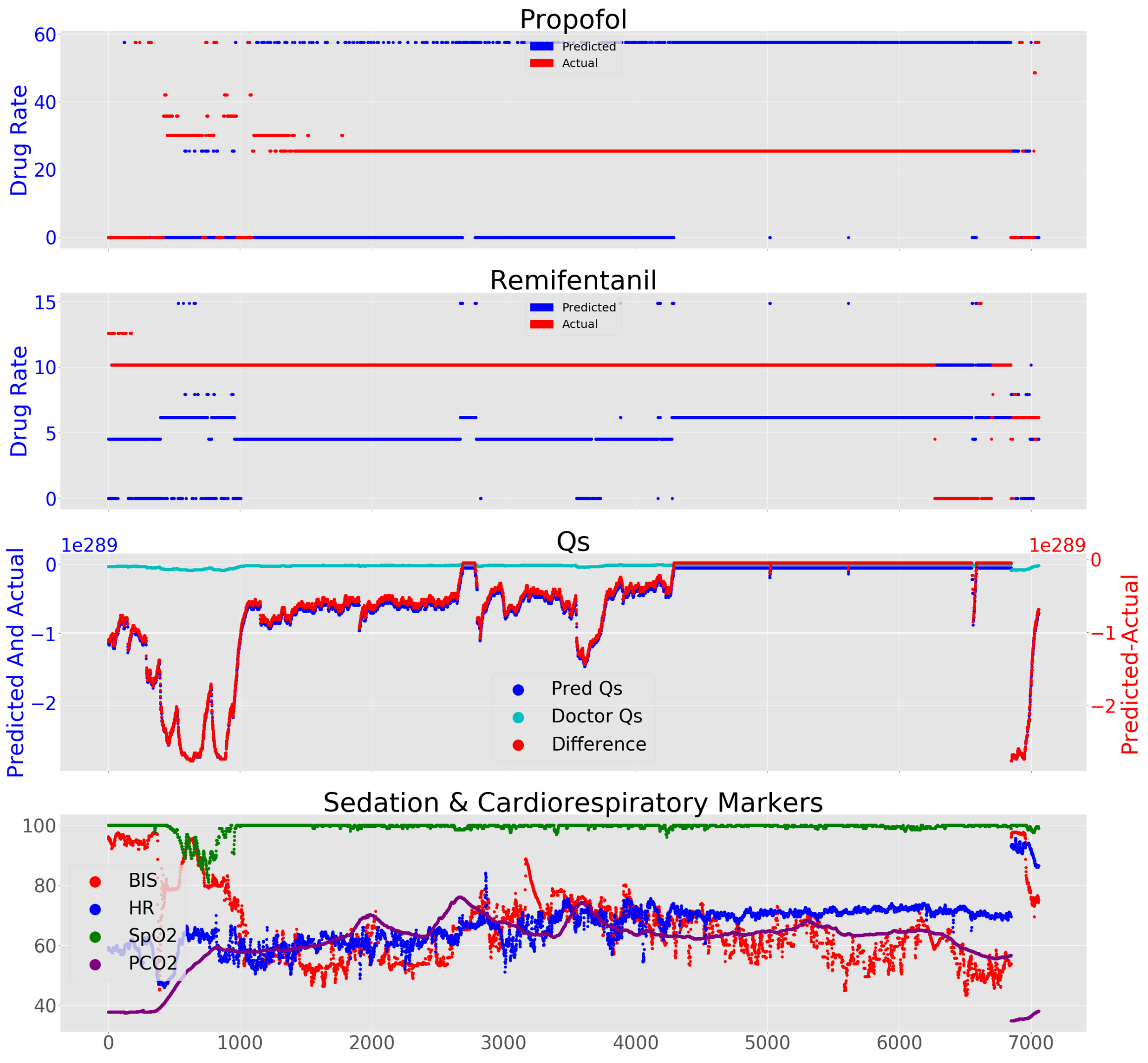


Figure 4-3: Linear Regressor on Single Timestep model with medians of predicted action bins plotted. Blue line corresponds to the RL actions, and red line corresponds to doctor's actions. This is for one patient.

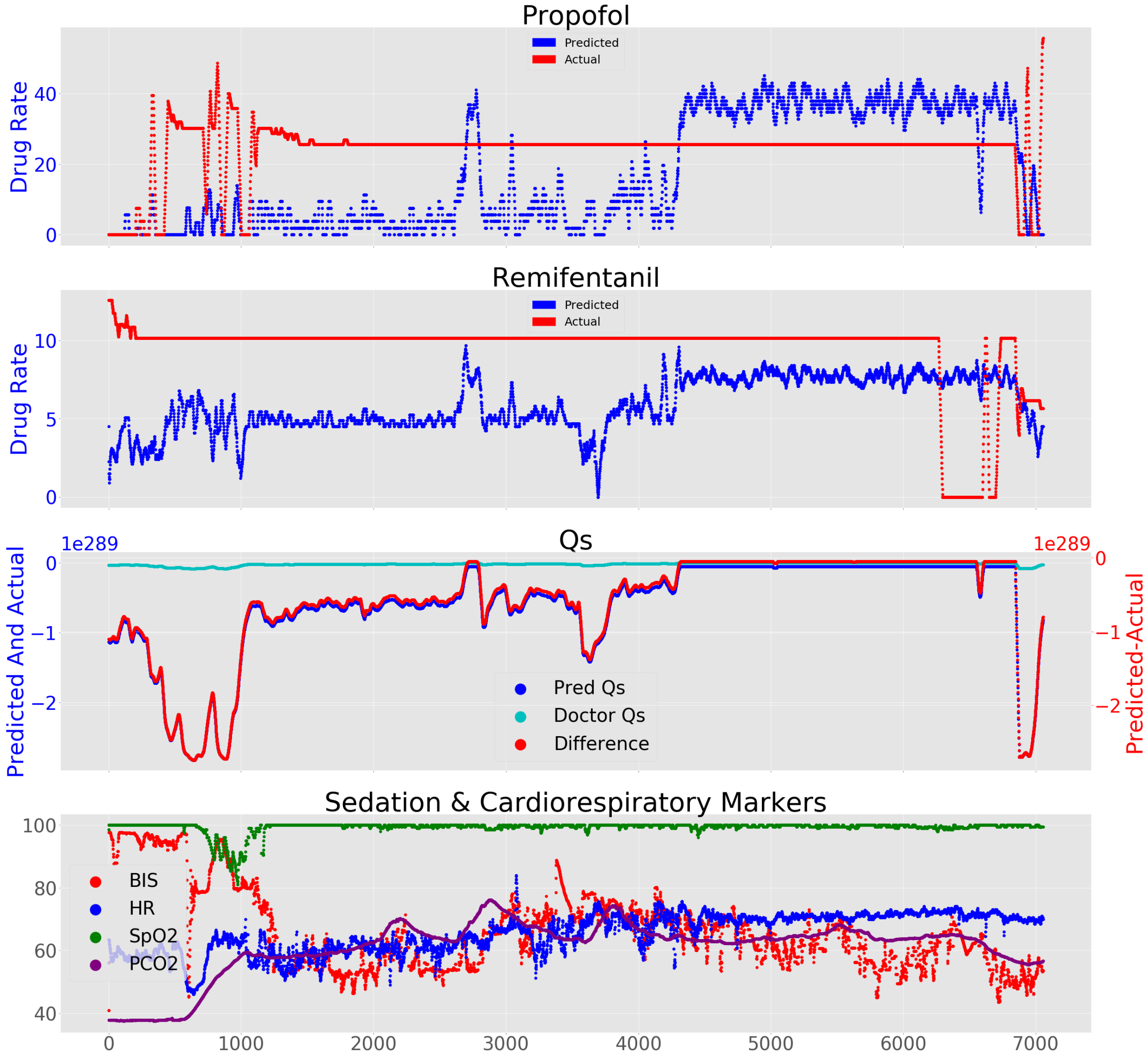


Figure 4-4: Linear Regressor on Single Timestep model with medians of predicted action bins plotted and data smoothed to 30s. Blue line corresponds to the RL actions, and red line corresponds to doctor's actions. This is for the same patient depicted in Fig. **??**

4.2 Rolling Window

In order to capture more information in a patient's state representation, I used a rolling window of 30s with 5 aggregate functions. I took mean, median, standard deviation, min, and max of the previous 30s of data at each timestep for all the variables listed in section 4.1 except the static variables (Height, Age, Gender, Weight) which were left unchanged to achieve a state space of 125 variables. The action space was kept the same as before with a one-hot encoded vector of 49 possible combinations of propofol and remifentanil. The reward function was calculated similarly with the minor change being that the 30s means of the reward variables were used to calculate the error instead of just the raw values. I ran this model setup with Linear Regressor and with Decision Trees Regressor for 200 iterations each. The following figures depict the policy (smoothed and unsmoothed) learned for one patient in the training set and one patient in the test set. An important point to note is that the patient for whom the policy is shown in Figure 4-5 and Figure 4-6 is the same patient from Figure 4-3 and Figure 4-4. Yet the policy learned by this rolling window model is much different than the policy learned by the single timestep model. In this policy, the RL model almost always doses 0 for except a few timesteps where it doses a high rate as shown in the unsmoothed Figure 4-5. In the smoothed version, the policy looks very close to 0 for the patient in the training set. For the patient in the test set, the model similarly underdoses when compared to the clinician but does have a few peaks that are at a higher magnitude than the values predicted for the patient in the training set. However, the differences in the Q-values between the clinician policy and RL policy seem to always be 0 or positive, indicating that the RL model consistently had higher or equivalent Q-values to the clinician. Note that the Pred Qs line was plotted, however it is not visible because Doctor Qs line is overlaid on top of it since the Q-values are so similar.

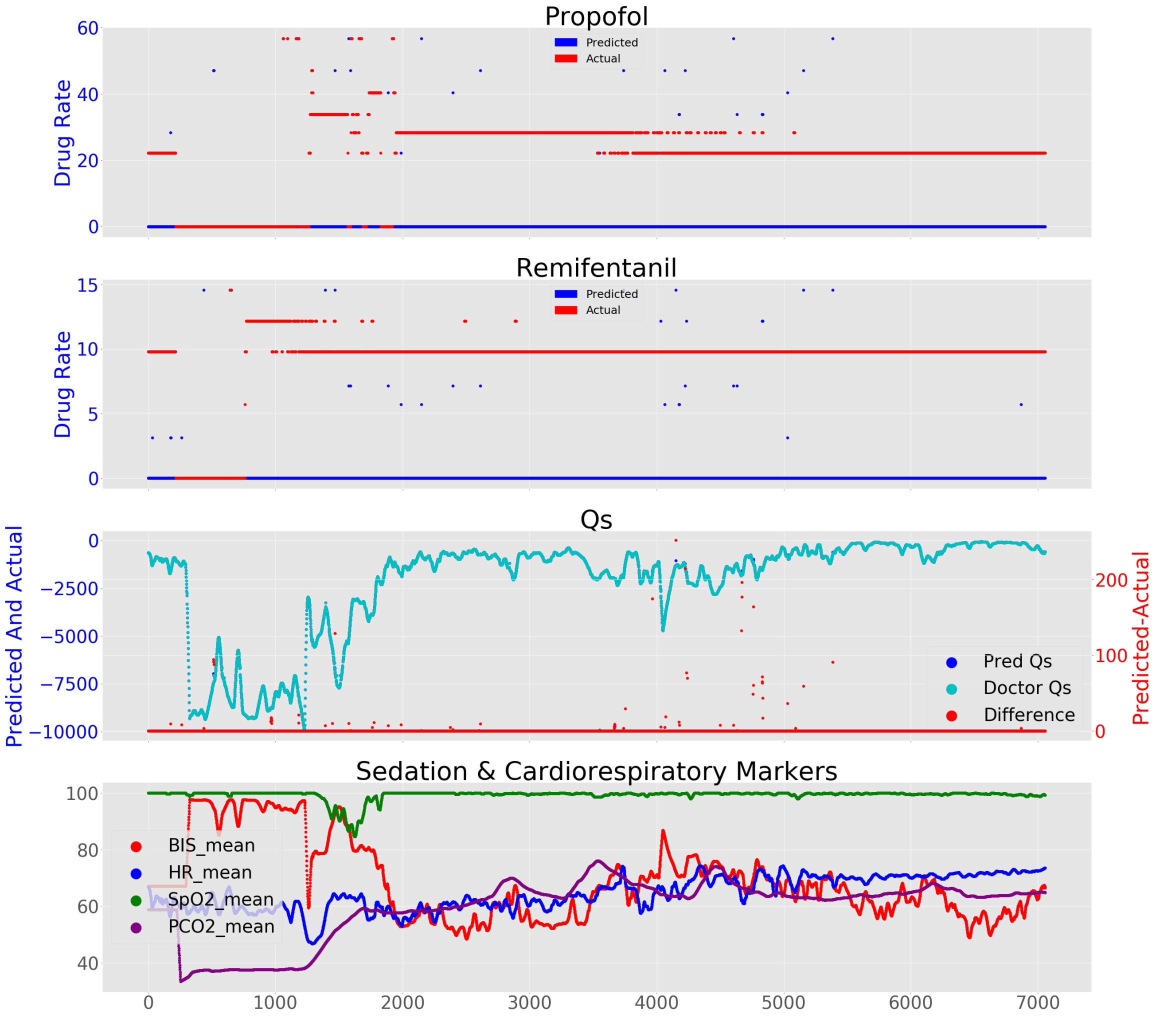


Figure 4-5: Policy for one patient under the rolling window model with decision trees regressor, unsmoothed.

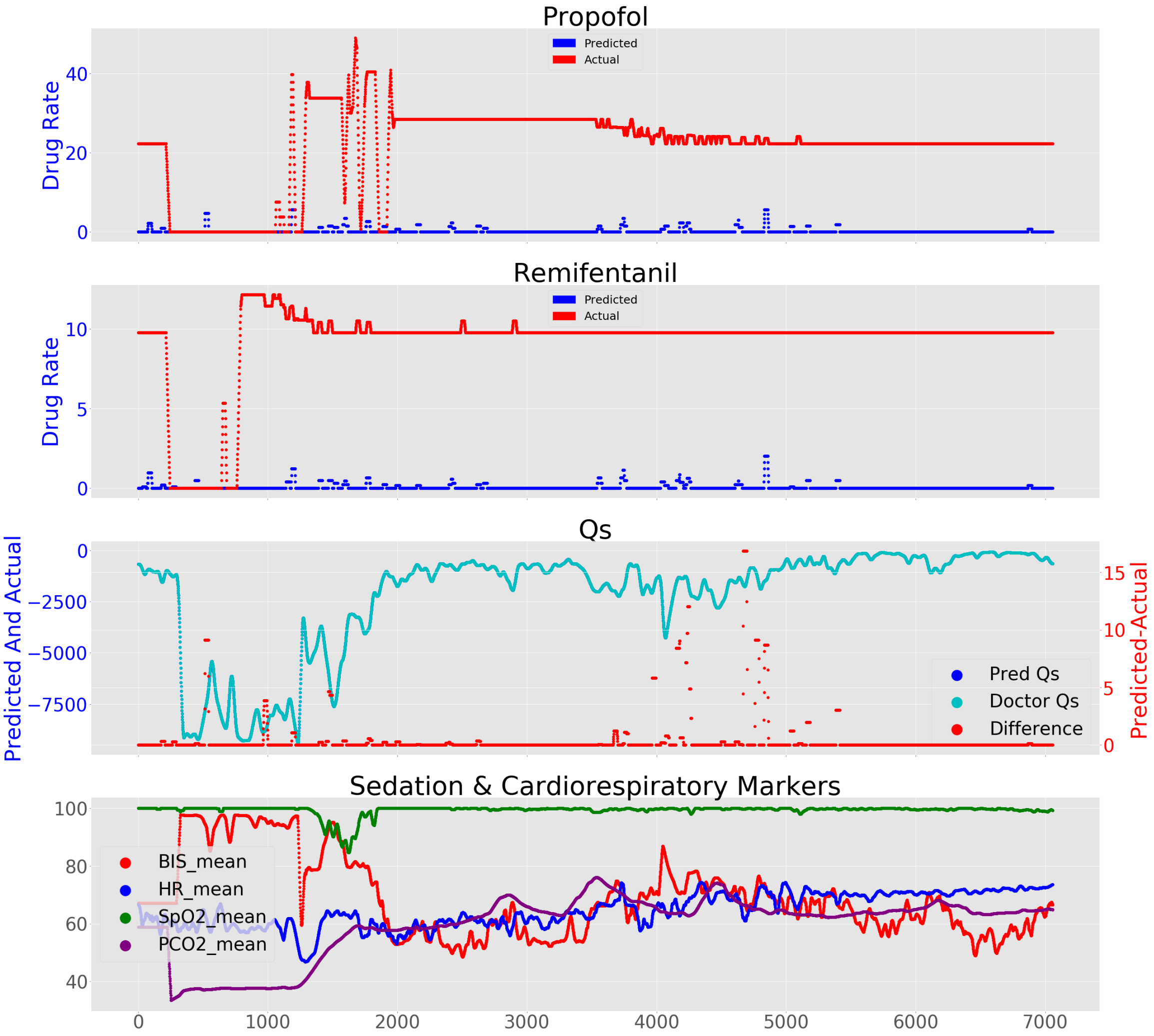


Figure 4-6: Policy for one patient under the rolling window model with decision trees regressor smoothed over 30s with a moving average.

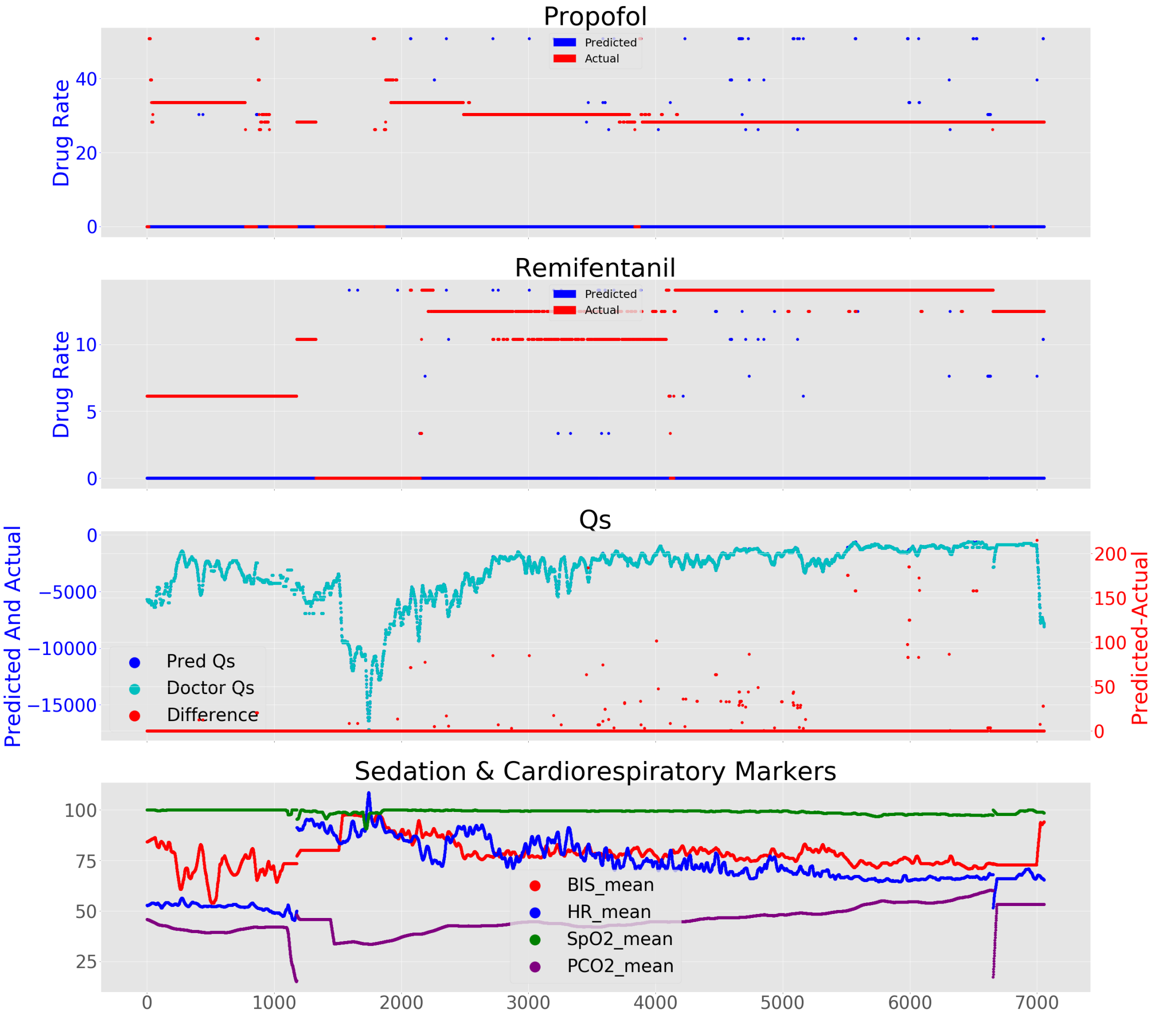


Figure 4-7: Policy for one patient in the test set under the rolling window model with decision trees regressor, unsmoothed.

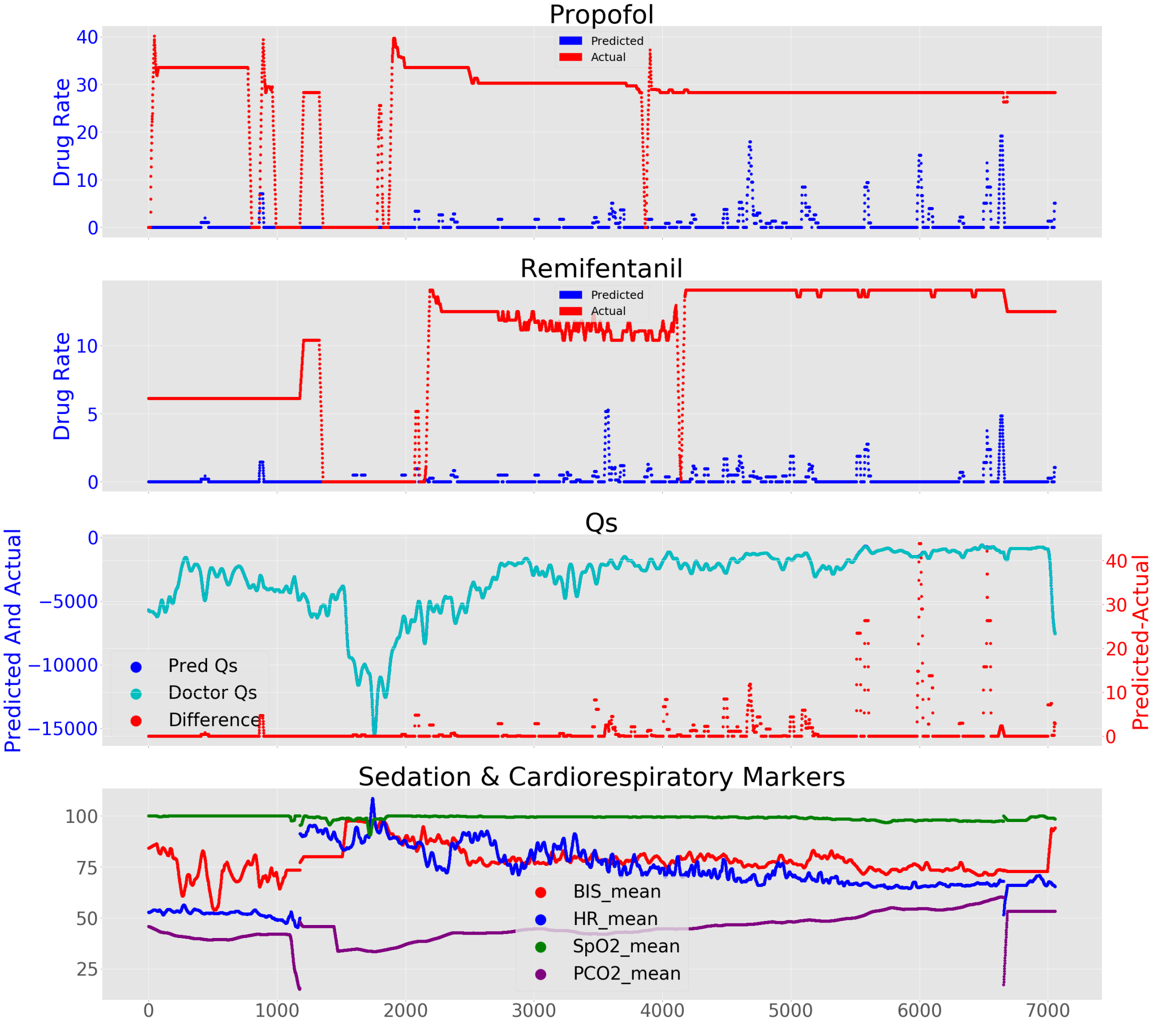


Figure 4-8: Policy for one patient in the test set under the rolling window model with decision trees regressor smoothed over a 30s with a moving average.

4.3 Stacked Window

The last setup that I tried is stacking model. Instead of a rolling function, I took all the data present in the 29 state vars over the last 15 seconds as for each timestep and concatenated it. The action space remained the same. The reward was slightly modified to penalize based on an individual patient's baseline heart rate instead of penalizing based on a general population baseline. I established baseline heart rates for each patient by averaging the heart rate over the first 5 seconds of a patient's stay in the hospital. I then applied the penalty in 3 tiers where if the patient's heart rate was within 10% of the baseline, the penalty was only 10% of the squared difference, if it was 20% within, the penalty was 20% of the squared difference, and full squared difference otherwise. This was a decision made from discussion with Dr. Pedro Gambus, our collaborator on this project, to adjust for each patient's natural heart rate instead of expecting every patient to be close to one normal value. I also made the target value for SpO2 slightly higher to be closer to what clinicians aim for.

```
def calc_reward(row,bh):## next state r1 based on s2
    rew = 0
    rew +=(row["BIS"]-57)**2
    rew +=(row["RSS"]-4)**2
    if row["PCO2"]>60:
        rew+=(row["PCO2"]-60)**2
    if row["SpO2"]<90:
        rew+=(90-row["SpO2"])**2
    if bh*.10<=abs(bh-row["HR"]):
        rew+=.10*(row["HR"]-bh)**2
    elif bh*.20<=abs(bh-row["HR"]):
        rew+=.20*(row["HR"]-bh)**2
    else:
        rew+=(row["HR"]-bh)**2
    return -rew
```

I ran this model for 100 iterations using a Linear Regressor with $\gamma = .8$. However the actions predicted were trivial and always the same for every timestep across all the patients–the model outputted 1 for propofol and 0 for remifentanil.

Chapter 5

Evaluation

The following analyses were performed on the results from the rolling window model with the decision tree regressor described in chapter 4. I chose this model to demonstrate the evaluation techniques but the methods are generalizable to results from any of the models outlined.

5.1 U-curve

First, I applied the U-curve method to qualitatively understand the RL policy. This method involves associating the learned RL policy to some outcome. Because the patients in this study were undergoing gastrointenstinal endoscopy, one of the main goals of sedation was to prevent patient gag reflex during the insertion of the endoscopic tube. To understand how RL predicted actions and clinician actions correspond to gag response, I first aggregated all the actions across a patient and then took a weighted average of the predicted bins to output 2 values for each patient, corresponding to the weighted average of RL actions and weighted average of clinician actions. The bin values correspond to the ranges outlines in Table 4.1. I plotted the results and color coded them based on whether the patient had gag reflex or not. As we can see in Fig. 5-1, the RL model predicted a much lower dosage on average than the doctor actually dosed. However, gag reflex also only occurred only 25% of the time, as depicted in green and cyan. Thus, it is possible that the clinician overdosed 75% of the time and the RL model underdosed 25% of the time. However, this cannot be concluded definitively because the fact that gag response happened at all means that even the clinician sometimes underdosed the patients. If the clinician, who consistently gave a higher dose than the RL model, underdosed, it is possible the RL model is not just undersoing 25% of the time but severely underdosing the patient most of the time.

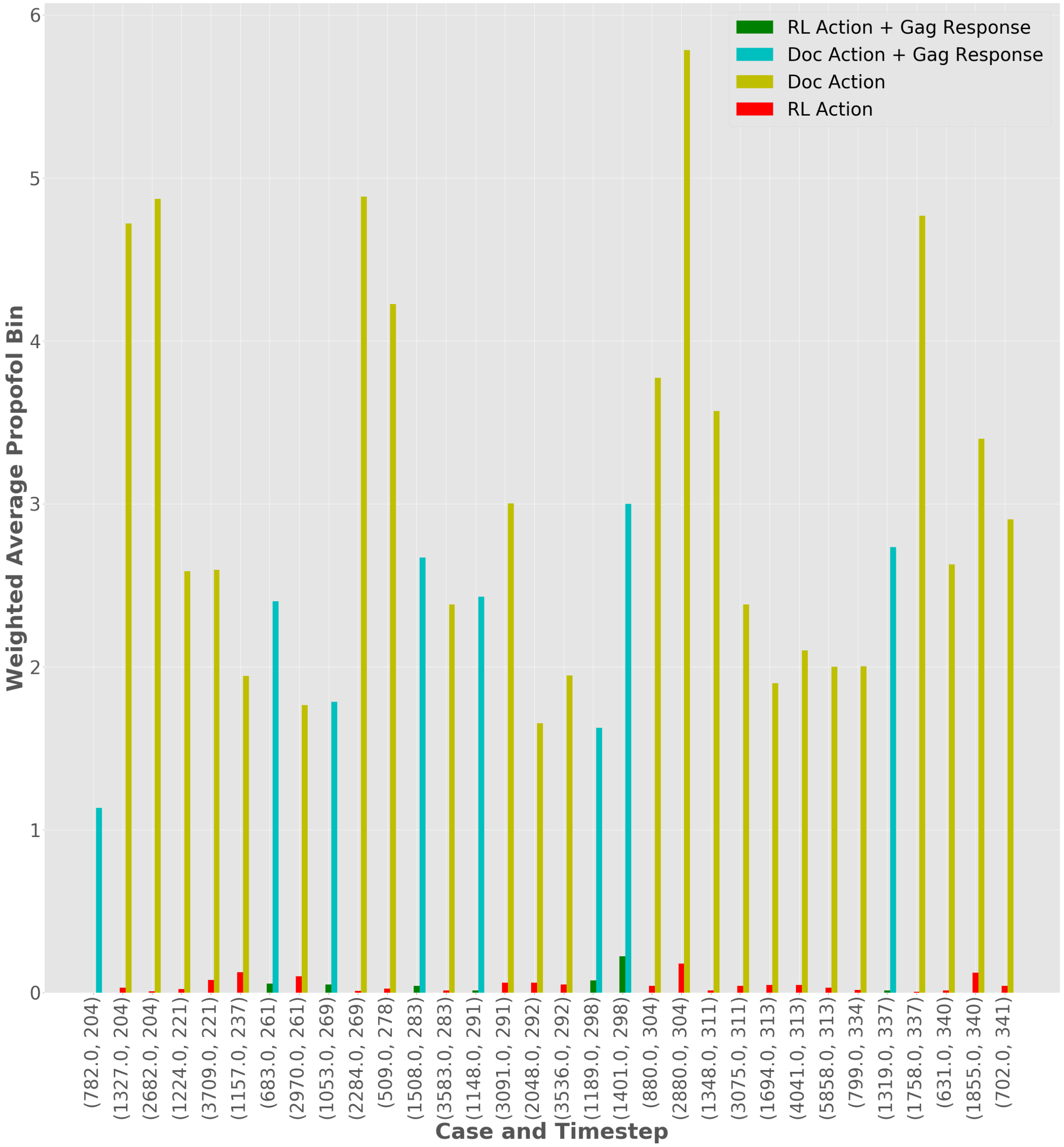


Figure 5-1: Plot of the average propofol action across each patient in the test set according to the RL model and doctor and whether gag response occurred.

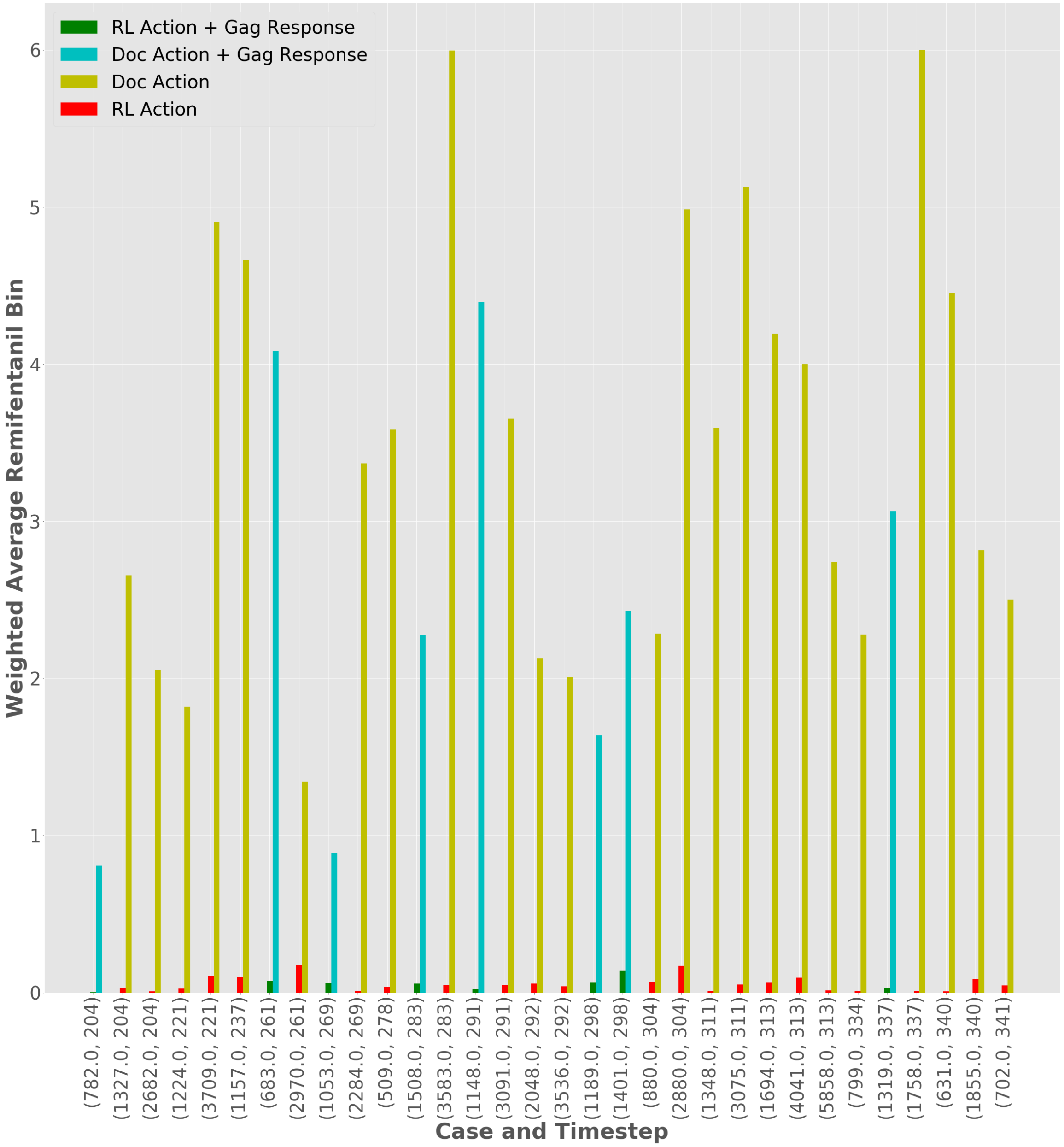


Figure 5-2: Plot of the average remifentanil action across each patient in the test set according to the RL model and doctor and whether gag response occurred. The green and cyan correspond to gag response occurring.

5.2 Doubly Robust Methods

I also implemented the doubly robust method in python to evaluate the RL policy. This statistical approach uses importance sampling to output a value for each patient of the policy learned. I first trained two state transition probability models (π_b, π_e) on the training dataset that was used to learn the RL policy with $x = \mathcal{S}$ and $y = \mathcal{A}$. It is both extremely crucial and incredibly challenging to build well-calibrated models that learn the behavior policy accurately in order to ensure the off-policy evaluation method works [24]. The RL policy was not hard to learn with the training and validation accuracies being 97.6% and 98.2%, suggesting that the model (π_e) was well-calibrated. However, the training and validation accuracies for the clinician policy (π_b) were 23.6% and 17.5%, suggesting that we were not able to learn the clinician policy from a simple decision tree regressor with as high accuracy as is necessary to ensure meaningful evaluation measures. Nevertheless, I implemented the evaluation technique using Algorithm 2 to calculate the an estimate of the value of the learned policy for each patient. Table 5.1 lists the value of the policy learned for each patient when γ was set equal to .05.

This procedure needs extensive tuning and analysis because the choice of γ, the choice of how much we choose to update V in each iteration, highly dictates the final calculated value of the policy. A high γ leads to an overflow error and and low γ leads to a negative value as seen in Table 5.1. Furthermore, in order to interpret the value outputted, we also need to calculate the value of the clinician policy for comparison. A more thorough analysis would be to also calculate the value of a random action policy and value of no action policy to use for comparison to the RL policy and determine if the RL policy in itself has a higher value or any policy would result in a high value due to the way we set up the MDP. In fact, Gottesman et al. showed that sometimes the way the action space is constructed can result in even a random action policy having a comparable value to a thoroughly trained RL model. This suggests that Importance Sampling methods may serve as a baseline, but alone are not complete methods of evaluation.

Patient	**Value**
1	30107.682411837362
2	17162.509723541363
3	-20419.88674567981
4	-17665.156992367843
5	-20419.88674567981
6	-19489.288370987233
7	-19512.96176763476
8	779327.4757294233
9	-17084.933502924225
10	-20224.95008121079
11	10476.65024905686
12	-20419.88674567981
13	-13314.727608279367
14	-17740.480048234553
15	-20419.88674567981
16	-19991.607450652602

Table 5.1: Value of policy learned by the rolling window model with decision tree regressor for each patient across the 16 patients in the test set.

5.3 Policy Discussion

Off-policy evaluation is a hard problem in the clinical setting. We don't want to just learn the policy the doctors followed since we want to do better yet we don't have model of all the possible transition dynamics for a patient. We only have the historical state, action pairs seen. This problem is exacerbated in Reinforcement Learning where Importance Sampling methods serve as a proxy in the absence of being able to actually deploy the evaluation policy in a simulation or a volunteer study. Moreover, the success of Importance Sampling methods greatly depends on the calibration of the models to the actual patterns underlying the data.

A problem with the U-curve method is that we cannot evaluate every decision made by the RL model at each timestep since the outcome measured only happens 1-2 times per patient. It is only possible to evaluate collections of decisions (the decisions made leading up to the point at which the outcome is measured), not each decision made by the RL model. Furthermore, one outcome may not be sufficient to evaluate the learned policy. In this work, we use gag response as the outcome.

However, gag response only serves as an indication of *underdosing* a patient. A patient may be *overdosed* but we have no exact outcome that measures this other interpreting the variables we used in the reward function to serve as indicators of hemodynamic depression (consequence of overdosing). However since the RL model was already trained to optimize for the control over these variables, the RL model should, by design, be better than the clinician policy and hence leads to a biased evaluation metric.

Chapter 6

Discussion and Future Work

Reinforcement Learning in a clinical setting is a tough problem because choosing the right state representation, action space, and reward function can greatly affect the policy learned by the model. In this work itself, with the same reward function and same state representation, an action space of 10 bins caused the model to output (0,0) as the action to take at every timestep due to the underlying distribution of drug values within each bin. Varying the state representation also drastically changed the learned policy. The stacked window model with 15s of data always outputted one set of actions whereas the other models outputted a varied set.

The state space representation is an extremely crucial piece of the puzzle. What constitutes an accurate state representation of a patient? In the clinical setting, doctors can see the entire history of a patient. However when we model this problem as an MDP, we make the assumption that only the current state is necessary to predict the action to take at the next timestep. Verifying that all the confounders are included in the state representation is also hard to do and just incorporating all the variables present in a dataset may only introduce noise into the model. There are many ways to represent a state and one option could be to use dimensionality reduction. Either clustering the state variables or applying PCA across a certain window of time might be work better.

Careful choice over the action space is also crucial. There is a balance to achieve between having too many options for the action space and having too few. Too few options and the model might recommend one action at a much higher rate than is needed because of a lack of other options to choose from [6]. Too many options and the problem may not be tractable for the RL model to converge. The approach I took in this thesis is to bin a continuous action space into discrete set of actions. However, the bins need to be sufficiently small for the policy to actually be clinically useful and in practice, doctors don't view sedation control as having to choose from a discrete set of bins. Other options for action spaces worth noting are discretizing the action space as {+1, -1, 0} where the model has to suggest whether to increase, decrease, or keep the rate the same at each state and using an action space that discretizes the entire range of possible values at a specified step size. For example, if the drug rate can range between 0 to 6, the action space could be all the possible values in that range at a step size of 0.1: {0,0.1, 0.2, 0.3,...,6.0}. At each state, the RL model would have to chose a rate from all possible rates in this set.

Furthermore, our current reward function is a simple mean squared error across key variables often used in sedation control to monitor for hemodynamic depression. However, this should not be the only goal of an RL model dosing sedation. Gag response and other EEG derived measures may be useful to achieve a more personalized regime for each patient. The reward function can also be determined with Inverse RL as described in Chapter 2 in order to learn the rewards from the data itself.

Finally the algorithm itself, FQI, can output different results based on the regressor used to learn the Q-value function. In this work, I tried LinearRegressor and DecisionTreeRegressor for the speed and simplicity. However, a neural network regressor may be faster and better able to represent the q-values since at each iteration, the model simply has to update the weights on the network, not retrain an entire network. Batch FQI algorithms are another options which train on subsets of the data instead of the entire dataset which might help in generalizing the Q-value function.

Ultimately, no matter what policy is learned, it is hard to evaluate it without being able to test the policy in either a simulated environment or in real life. A potential future direction of this work may be to extensively focus on building an accurate simulation of a patient undergoing gastrointenstinal endoscopy and then test various

RL models and their learned policies to understand how different setups affect the learned policies.

Appendix A

Tables

Table A.1: Acronyms used in the dataset

Acronyms	**Variable Definition**
AAI	Auditory evoked potential index
BIS	Bispectral Index
BSA	Body Surface Area
BSAAI	Brain stem derived Auditory evoked potential index
BSBIS	Brain stem derived Bispectral index
CePROPO	Effect site concentrations of propofol
CeREMI	Effect site concentrations of remifentanil
CpPROPO	Plasma concentrations of propofol
CpREMI	Plasma concentrations of remifentanil
EMGAAI	EMG derived Auditory evoked potential index
EMGBIS	EMG derived Bispectral index
GABRB3	Flag for genetic mutation associated with sedation response
GAG	Measure of gag response when patient is intubated
HR	Heart rate continuously measured
InfRatePROPO	Continuous infusion rate of propofol
InfRateREMI	Continuous infusion rate of remifentanil
InfVolPROPO	Total infusion volume of propofol
InfVolREMI	Total infusion volume of remifentanil
LBM	Lean Body Mass
NIBPdia	Noninvasive diastolic blood pressure
NIBPmean	Noninvasive mean blood pressure
NIBPsys	Noninvasive systolic blood pressure
OPRM1	Screening for A118G gene
PCO2	Partial pressure of carbon dioxide
RSS	Ramsay sedation score
RespiRate	Respiration Rate
SQI09	Signal Quality Index
TUBE	Binary flag whether the patient is intubated
SpO2	Oxygen saturation

Bibliography

[1] Ngai American Society of Anesthesiologists., Thierry Chazot, Antoine Genty, Alain Landais, Aymeric Restoux, Kathleen McGee, Pierre-Antoine Laloë, Bernard Trillat, Luc Barvais, and Marc Fischler. *Anesthesiology.*, volume 104. [American Society of Anesthesiologists, etc.], 4 2006.

[2] Xavier Borrat, Marta Ubre, Raquel Risco, Pedro L. Gambús, Angela Pedroso, Aina Iglesias, Gloria Fernandez-Esparrach, ngels Ginés, Jaume Balust, and Graciela Martínez-Palli. Computerized tests to evaluate recovery of cognitive function after deep sedation with propofol and remifentanil for colonoscopy. *Journal of Clinical Monitoring and Computing*, 0(0):1–7, 2 2018.

[3] Olivier Caelen, Olivier Cailloux, Djamal Ghoundiwal, and Abhilash Alexander. Real-time prediction of an anesthetic monitor index using machine learning. (August 2014), 2011.

[4] P. L. Gambús, E. W. Jensen, M. Jospin, X. Borrat, G. Martnez Pallí, J. Fernández-Candil, J. F. Valencia, X. Barba, P. Caminal, and I. F. Trocóniz. Modeling the Effect of Propofol and Remifentanil Combinations for Sedation-Analgesia in Endoscopic Procedures Using an Adaptive Neuro Fuzzy Inference System (ANFIS). *Anesthesia & Analgesia*, 112(2):331–339, 2 2011.

[5] Pedro L Gambús and Iaki F Trocóniz. Pharmacokineticpharmacodynamic modelling in anaesthesia. *British Journal of Clinical Pharmacology*, 79(1):72, 1 2015.

[6] Omer Gottesman, Fredrik Johansson, Matthieu Komorowski, Aldo Faisal, David Sontag, Finale Doshi-Velez, and Leo Anthony Celi. Guidelines for reinforcement learning in healthcare. *Nature Medicine*, 25(1):16–18, 2019.

[7] Wassim M. Haddad, James M. Bailey, Behnood Gholami, and Allen R. Tannenbaum. Clinical Decision Support and Closed-Loop Control for Intensive Care Unit Sedation. *Asian Journal of Control*, 15(2):317–339, 3 2013.

[8] J. A. Hannam, X. Borrat, I. F. Troconiz, J. F. Valencia, E. W. Jensen, A. Pedroso, J. Munoz, S. Castellvi-Bel, A. Castells, and P. L. Gambus. Modeling Respiratory Depression Induced by Remifentanil and Propofol during Sedation and Analgesia Using a Continuous Noninvasive Measurement of pCO2. *Journal of Pharmacology and Experimental Therapeutics*, 356(3):563–573, 1 2016.

[9] Nan Jiang and Lihong Li. Doubly Robust Off-policy Value Evaluation for Reinforcement Learning. 48, 2015.

[10] G N Kenny and H Mantzaridis. Closed-loop control of propofol anaesthesia. *British Journal of Anaesthesia*, 83(2):223–228, 8 1999.

[11] Stephan C Kettner. Not too Little, not too Much: Delivering the Right Amount of Anaesthesia during Surgery. *Cochrane Database of Systematic Reviews*, pages 8–9, 2014.

[12] Hyung-Chul Lee, Ho-Geol Ryu, Eun-Jin Chung, and Chul-Woo Jung. Prediction of Bispectral Index during Target-controlled Infusion of Propofol and Remifentanil. *Anesthesiology*, 128(3):492–501, 3 2018.

[13] N. Liu, C. Lory, V. Assenzo, V. Cocard, T. Chazot, M. Le Guen, D.I. Sessler, D. Journois, and M. Fischler. Feasibility of closed-loop co-administration of propofol and remifentanil guided by the bispectral index in obese patients: a prospective cohort comparison . *British Journal of Anaesthesia*, 114(4):605–614, 4 2015.

[14] Ngai Liu, Thierry Chazot, Antoine Genty, Alain Landais, Aymeric Restoux, Kathleen Mcgee, Pierre-Antoine Laloë, Bernard Trillat, Luc Barvais, and Marc Fischler. Titration of Propofol for Anesthetic Induction and Maintenance Guided by the Bispectral Index: Closed-loop versus Manual Control A Prospective, Randomized, Multicenter Study. Technical report, 2006.

[15] Ngai Liu, Morgan Le Guen, Fatima Benabbes-Lambert, Thierry Chazot, Bernard Trillat, Daniel I. Sessler, and Marc Fischler. No Title, 2 2012.

[16] Mahdi Mahfouf, Catarina S. Nunes, Derek A. Linkens, and John E. Peacock. Modelling and multivariable control in anaesthesia using neural-fuzzy paradigms: Part II. Closed-loop control of simultaneous administration of propofol and remifentanil. *Artificial Intelligence in Medicine*, 35(3):207–213, 11 2005.

[17] Umberto Melia, Montserrat Vallverdú, Xavier Borrat, Jose Fernando Valencia, Mathieu Jospin, Erik Weber Jensen, Pedro Gambus, and Pere Caminal. Prediction of nociceptive responses during sedation by linear and non-linear measures of EEG signals in high frequencies. *PLoS ONE*, 10(4):1–21, 2015.

[18] Brett L Moore, Larry D Pyeatt, Vivekanand Kulkarni, Periklis Panousis, and Kevin Padrez. Reinforcement Learning for Closed-Loop Propofol Anesthesia: A Study in Human Volunteers. Technical report, 2014.

[19] Sunil B. Nagaraj, Lauren M. McClain, David W. Zhou, Siddharth Biswal, Eric S. Rosenthal, Patrick L. Purdon, and M. Brandon Westover. Automatic Classification of Sedation Levels in ICU Patients Using Heart Rate Variability. *Critical care medicine*, 44(9):782–9, 9 2016.

[20] Shamim Nemati, Mohammad M Ghassemi, and Gari D Clifford. Optimal medication dosing from suboptimal clinical examples: a deep reinforcement learning approach. *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference*, 2016:2978–2981, 8 2016.

[21] Regina Padmanabhan, Nader Meskin, and Wassim M. Haddad. Reinforcement learning-based control for combined infusion of sedatives and analgesics. In *2017 4th International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 0505–0509. IEEE, 4 2017.

[22] Niranjani Prasad, Li-Fang Cheng, Corey Chivers, Michael Draugelis, and Barbara E Engelhardt. A Reinforcement Learning Approach to Weaning of Mechanical Ventilation in Intensive Care Units. 4 2017.

[23] A Raghu, M Komorowski, L A Celi, and P Szolovits. Continuous State-Space Models for Optimal Sepsis Treatment-a Deep Reinforcement Learning Approach. *arXiv.org*, 1 2017.

[24] Aniruddh Raghu, Omer Gottesman, Yao Liu, Matthieu Komorowski, Aldo Faisal, Finale Doshi-Velez, and Emma Brunskill. Behaviour Policy Estimation in Off-Policy Policy Evaluation: Calibration Matters. Technical report, 2018.

[25] DharmaJivan Samantaray, Meena Trehan, Vivek Chowdhry, and Satish Reedy. Comparison of hemodynamic response and postoperative pain score between general anaesthesia with intravenous analgesia versus general anesthesia with caudal analgesia in pediatric patients undergoing open-heart surgery. *Annals of Cardiac Anaesthesia*, 22(1):35, 2019.

[26] S L Shafer and K M Gregg. Algorithms to rapidly achieve and maintain stable drug concentrations at the site of drug effect with a computer-controlled infusion pump. *Journal of pharmacokinetics and biopharmaceutics*, 20(2):147–69, 4 1992.

[27] Christopher J.C.H. Watkins and Peter Dayan. Technical Note: Q-Learning. *Machine Learning*, 8(3):279–292, 1992.

[28] Chao Yu, Jiming Liu, and Hongyi Zhao. Inverse reinforcement learning for intelligent mechanical ventilation and sedative dosing in intensive care units. *BMC Medical Informatics and Decision Making*, 19(Suppl 2), 2019.