DEEP RL FOR BLOOD GLUCOSE CONTROL: LESSONS, CHALLENGES, AND OPPORTUNITIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Individuals with type 1 diabetes (T1D) lack the ability to produce the insulin their bodies need. As a result, they must continually make decisions about how much insulin to self-administer in order to adequately control their blood glucose levels. Longitudinal data streams captured from wearables, like continuous glucose monitors, can help these individuals manage their health, but currently the majority of the decision burden remains on the user. To relieve this burden, researchers are working on closed-loop solutions that combine a continuous glucose monitor and an insulin pump with a control algorithm in an 'artificial pancreas.' Such systems aim to estimate and deliver the appropriate amount of insulin. Here, we develop reinforcement learning (RL) techniques for automated blood glucose control. Through a series of experiments, we compare the performance of different deep RL approaches to non-RL approaches. We highlight the flexibility of RL approaches, demonstrating how they can adapt to new individuals with little additional data. On over 21k hours of simulated data across 30 patients, RL approaches outperform baseline control algorithms (increasing time spent in normal glucose range from 71% to 75%) without requiring meal announcements. Moreover, these approaches are adept at leveraging latent behavioral patterns (increasing time in range from 58% to 70%). This work demonstrates the potential of deep RL for controlling complex physiological systems with minimal expert knowledge.

1 INTRODUCTION

Type 1 diabetes (T1D) is a chronic disease affecting 20-40 million people worldwide (You & Henneberg, 2016), and its incidence is increasing (Tuomilehto, 2013). People with T1D cannot produce insulin, a hormone that signals cells to uptake glucose in the bloodstream. Without insulin, the body must metabolize energy in other ways that, when relied on repeatedly, can lead to life-threatening conditions (Kerl, 2001). Tight glucose control improves both short- and long-term outcomes for people with diabetes, but can be difficult to achieve in practice (Diabetes Control and Complications Trial Research Group, 1995). Typically, blood glucose is controlled by a combination of basal insulin (to control baseline blood glucose levels) and bolus insulin (to control glucose spikes after meals). To control blood glucose levels, individuals with T1D must continually make decisions about how much basal and bolus insulin to self-administer. This requires careful measurement of glucose levels and carbohydrate intake, resulting in at least 15-17 data points a day. If the individual uses a continuous glucose monitor (CGM), this can increase to over 300 data points, or a blood glucose reading every 5 minutes (Coffen & Dahlquist, 2009).

Combined with an insulin pump, a wearable device that automates the delivery of insulin, CGMs present an opportunity for closed-loop control. Such a system, known as an 'artificial pancreas' (AP), automatically anticipates the amount of required insulin and delivers the appropriate dose. This would be life-changing for individuals with T1D. For many years, researchers have worked towards the creation of an AP for blood glucose control (Kadish, 1964; Bequette, 2005; Bothe et al., 2013). Though the technology behind CGMs and insulin pumps has advanced, there remains significant room for improvement when it comes to the control algorithms (Bothe et al., 2013; Pinsker et al., 2016). Current approaches often fail to maintain sufficiently tight glucose control and require meal announcements.

In this work, we investigate the utility of a deep reinforcement learning (RL) based approach for blood glucose control (Bothe et al., 2013). Deep RL is particularly well-suited for this task because it: i) makes minimal assumptions about the structure of the underlying process, allowing the same system to adapt to different individuals or to changes in individuals over time, ii) can learn to leverage latent patterns such as regular meal times, and iii) scales well in the presence of large amounts of training data. Finally, it can take advantage of existing FDA-approved simulators for model training. Despite these potential benefits, we are not aware of any previously published work that has rigorously explored the feasibility of deep RL for blood glucose control. Thus, we present the first large scale evaluation of such an approach, demonstrating that, despite a number of technical challenges, deep RL can be used to learn good AP algorithms.

2 Background and Related Works

In recent years, researchers have started to explore RL in healthcare. Examples include matching patients to treatment in the management of sepsis (Weng et al., 2017; Komorowski et al., 2018) and mechanical ventilation (Prasad et al., 2017). In addition, RL has been explored to provide contextual suggestions for behavioral modifications (Klasnja et al., 2019). Despite its success in other problem settings, RL has yet to be fully explored as a solution for a closed-loop AP system (Bothe et al., 2013). RL is a promising solution to this problem, as it is well-suited to learning complex behavior that readily adapts to changing domains (Clavera et al., 2018). Moreover, unlike many other disease settings, there exist credible simulators for the glucoregulatory system (Visentin et al., 2014). The presence of a credible simulator alleviates many common concerns of RL applied to problems in health (Gottesman et al., 2019).

2.1 Current AP algorithms and RL for Blood Glucose Control

Among recent commercial AP products, proportional-integral-derivative (PID) control is one of the most common backbones (Trevitt et al., 2015). The simplicity of PID controllers make them easy to use, and in practice they achieve strong results. For example, the Medtronic Hybrid Closed-Loop system, one of the few commercially available, is built on a PID controller (Garg et al., 2017; Ruiz et al., 2012). In this setting, a hybrid closed-loop controller automatically adjusts basal insulin rates, but still requires human-directed insulin boluses to adjust for meals. The main weakness of PID controllers, in the setting of blood glucose control, is their reactivity. As they only respond to current glucose values (including a derivative), often they cannot respond fast enough to meals to satisfactorily control postprandial excursions without meal announcements (Garg et al., 2017). And, without additional safety modifications can overcorrect for these spikes, triggering postprandial hypoglycemia (Ruiz et al., 2012). In contrast, we hypothesize that an RL approach will be able to leverage patterns associated with meal times, resulting in better policies that do not require meal announcements. Moreover, such approaches can take advantage of existing simulators for training and evaluation (described in more detail later).

Weng et al. (2017) use RL to learn policies that set blood glucose targets for septic patients, but do not learn policies to achieve these targets. Most similar to our own work, De Paula et al. (2015) develop a kernelized Q-learning framework for closed loop glucose control (De Paula et al., 2015). They make use of Bayesian active learning for on-the-fly personalization. This work tackles a similar problem to our own, but uses a simple two-compartment model for the glucoregulatory system and a fully deterministic meal routine. In our simulation environment, we found that such a Q-learning did not lead to satisfactory closed-loop performance and instead we examine deep actor-critic algorithms for continuous control.

2.2 Glucose Models and Simulation

Models of the glucoregulatory system have long been important to the development and testing of an AP (Cobelli et al., 1982). Current models are based on a combination of rigorous experimentation and expert knowledge of the underlying physiological phenomena. Typical models consist of a multi-compartment model, with various sources and sinks corresponding to physiological phenomena, involving often dozens of patient-specific parameters. One such simulator, the one we use in our experiments, is the UVA/Padova model (Kovatchev et al., 2009). Briefly, this simulator models the glucoregulatory system as a nonlinear multi-compartment system, where glucose is generated through the liver and absorbed through the gut and controlled by externally administered insulin. A more detailed explanation can be found in (Kovatchev et al., 2009). We use an open-source version of the UVA/Padova simulator that comes with 30 virtual patients, each of which consists of several dozen parameters fully specifying the glucoregulatory system (Xie, 2018). The patients are divided into three classes: children, adolescents, and adults, each with 10 patients.

3 METHODS

The use of deep RL for blood glucose control presents several challenges. Through extensive experimentation, we found that the choice of state representation, action space, and reward function have significant impact on training and validation performance. Additionally, the high sample complexity of standard RL approaches for continuous control tasks can make the application of these methods in real-world settings infeasible. We address these challenges in turn, developing a learning pipeline that achieves strong performance across 30 different patients with the same architecture and hyperparameters without requiring meal announcements. Finally, we demonstrate how such policies can be transferred across patients in a data-efficient manner.

We begin by formalizing the problem. We then describe deep RL approaches that vary in terms of architecture and state representation, and present several baselines: an analogue to human-control in the form of a basal-bolus controller and variants on a PID controller.

3.1 PROBLEM SETUP

We frame the problem of blood glucose control as a Markov decision process (MDP) consisting of the 4-tuple (S, A, P, R). Our precise formulation of this problem varies depending on the method and setting. Here, we describe the standard formulation, and explain further differences as they arise. States $\mathbf{s}_t \in S$ consist of the previous 4 hours of blood glucose and insulin data at the resolution of 5-minute intervals: $\mathbf{s}_t = [\mathbf{b}^t, \mathbf{i}^t]$ where:

$$\mathbf{b}^t = [b_{t-47}, b_{t-46}, \dots b_t], \mathbf{i}^t = [i_{t-47}, i_{t-46}, \dots i_t]$$

and $b_t \in \mathcal{N}_{40:400}$, $i_t \in \mathcal{R}_{\geq 0}$, $t \in \mathcal{N}_{1:288}$ and represents a time index for a day at 5-minute resolution. We explored both longer (24 hours) and shorter (1-2 hours) length history as input, but after tuning on the validation data found that 4 hours struck a good balance between time to convergence and strong performance. We use an update resolution of 5 minutes to mimic the sampling frequency of many common continuous glucose monitors.

Actions $a_t \in \mathcal{R}_{\geq 0}$ are real positive numbers, denoting the size of the insulin bolus in medication units. We experimented with discritized action spaces (as is required by Q-learning approaches), but found such an approach lacked robustness across different discretization schemes. The transition function P consists of two elements: i) $G : (a_t, c_t) \rightarrow (b_{t+1}, i_{t+1})$, where $c_t \in \mathcal{R}_{\geq 0}$ is the amount of carbohydrates input at time t and G is a model of the glucoregulatory system, its behavior is defined in accordance with the UVA/Padova simulator (Kovatchev et al., 2009), ii) $M : t \rightarrow c_t$ is the meal schedule, and is defined in **Appendix A.1**.

The reward function R is defined as negative risk $-risk(b_t)$ where $risk$ is the asymmetric blood glucose risk function defined as:

$$risk(b) = 10 * (1.509 * \log(b)^{1.084} - 5.381)$$

shown in **Figure 1**, and is an established tool for computing glucose risk (Clarke & Kovatchev, 2009). We investigated using other reward functions, such as time in range or distance from a target blood glucose value, but found that optimizing for this reward function consistently led to better control.

3.2 SOFT ACTOR CRITIC

Our RL controller is trained using the Soft Actor Critic algorithm (Haarnoja et al., 2018). This algorithm is a natural choice for an AP algorithm, as it has been shown to be a reasonably sample efficient and well-performing algorithm when learning continuous control policies. This approach,

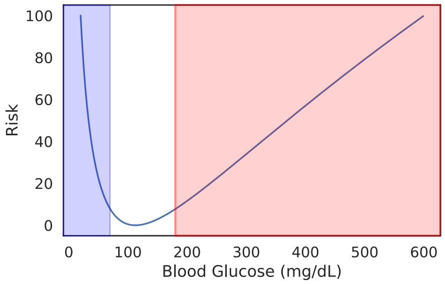


Figure 1: The risk function proposed in (Clarke & Kovatchev, 2009). The mapping between blood glucose values (in mg/dL, x-axis) and Risk values (y-axis). The hypo- and hyperglycemic thresholds are shown as shaded regions. The risk at the threshold of each region is approximately 7.75.

a member of the Actor-Critic family of algorithms, trains a stochastic policy network (or actor) parameterized by ϕ via to maximize the Maximum Entropy RL objective function:

$$J(\pi) = \sum_{0}^{T} \mathbb{E}_{(\mathbf{s}_t, a_t) \sim P(s_{t-1}, \pi_\phi(s_{t-1}))}[R(\mathbf{s}_t, a_t) + \alpha H(\pi_\phi(\cdot|\mathbf{s}_t))],$$

where the entropy regularization term, H, added to the expected cumulative reward improves exploration and robustness. This objective function is optimized by minimizing the KL divergence between the action distribution and the distribution induced by state-action values:

$$J_\pi(\phi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}}\left[\mathrm{D}_{\mathrm{KL}}\left(\pi_\phi\left(\cdot|\mathbf{s}_t\right) \| \frac{\exp\left(Q_\theta\left(\mathbf{s}_t, \cdot\right)\right)}{Z_\theta\left(\mathbf{s}_t\right)}\right)\right]$$

where $\mathcal{D}$ is a replay buffer containing previously seen $(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}_{t+1})$ tuples, Z_θ is a partition function, and Q_θ is the state-action value function parameterized by a neural network (also called a critic) and trained by minimizing the temporal difference loss:

$$J_Q(\theta) = \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \mathcal{D}}\left[\frac{1}{2}\left(Q_\theta\left(\mathbf{s}_t, \mathbf{a}_t\right) - \hat{Q}\left(\mathbf{s}_t, \mathbf{a}_t\right)\right)^2\right],$$

$$\hat{Q}\left(\mathbf{s}_t, \mathbf{a}_t\right) = r\left(\mathbf{s}_t, \mathbf{a}_t\right) + \gamma \mathbb{E}_{\mathbf{s}_{t+1} \sim p}\left[V_{\overline{\psi}}\left(\mathbf{s}_{t+1}\right)\right].$$

V_ψ is the soft value function parameterized by a third neural network, trained to minimize:

$$J_V(\psi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}}\left[\frac{1}{2}\left(V_\psi\left(\mathbf{s}_t\right) - \mathbb{E}_{\mathbf{a}_t \sim \pi_\rho}\left[Q_\theta\left(\mathbf{s}_t, \mathbf{a}_t\right) - \log \pi_\phi\left(\mathbf{a}_t|\mathbf{s}_t\right)\right]\right)^2\right],$$

and $V_{\overline{\psi}}$ is the running exponential average of the weights of V_ψ over training (a continuous variant of the hard target network replication in (Mnih et al., 2015)). Additional details of this approach, including the gradient calculations, are given in (Haarnoja et al., 2018). Note that we replace the MSE temporal difference loss with Huber loss, as we find this improves convergence.

3.2.1 Recurrent Architecture

Our proposed approach takes as input only the past 4 hours of CGM and insulin data, mimicking real-world applications without human input (*i.e.*, no meal announcements). To extract useful state information from the noisy CGM and insulin history, we parameterize Q_θ, V_ψ, and π_ϕ using GRU networks (Cho et al., 2014), as these types of architectures have successfully been used to model to blood glucose data in the past (Fox et al., 2018; Zhu et al., 2018). The GRU in π_ϕ maps states to a normal distribution $N(\mu, \log(\sigma))$, from which actions are sampled.

Given that one of the main disadvantages of RL approaches is their sample efficiency, we sought to explore transfer learning techniques that could allow networks trained from scratch to be efficiently transferred to new patients. We refer to our method trained from scratch as SAC-GRU, and the transfer approach as SAC-GRU-Trans. For SAC-GRU-Trans, we initialize $Q_\theta, V_\psi, \pi_\phi$ for each class of patients (children, adolescents, and adults) using fully trained networks from one randomly selected member of that source population (*e.g.*, Child/Adolescent/Adult 1). We then fine-tune these networks on data collected from the target patient. This provides a simple approach for training policies with potentially far less data per-patient.

Our GRU networks are two layers and have a hidden state size of 128, followed by a fully-connected output layer. Actions are squashed using a tanh function, and scaled by a parameter $\omega_b = 43.2 * bas$ where bas is the suggested basal insulin rate (which varies per-person). This scaling ensures that the maximum amount of insulin deliverable over a five minute interval is roughly equal to a normal meal bolus (Kuroda et al., 2011).

3.2.2 Oracle Architecture

A deep RL approach to learning AP algorithms requires that: i) the representation learned by the network contain sufficient information to control the system, and ii) an appropriate control algorithm be learned through interaction with the glucoregulatory system. As we are working with a simulator, we can explore the difficulty of task (ii) in isolation, by replacing the state $\mathbf{s}_t$ with the ground-truth state of the simulator $\mathbf{s}_t^*$, a 13-dimensional vector with real-valued elements representing glucose, carbohydrate, and insulin values in different compartments of the body. Though unavailable in real-world settings, this representation decouples the problem of learning a policy from that of learning a good state representation. Here, Q_θ, V_ψ, and π_ϕ are fully-connected with two hidden layers, each with 256 units. The network uses ReLU nonlinearities and BatchNorm (Ioffe & Szegedy, 2015).

3.3 Baselines

We examine three baseline methods for control: basal-bolus (BB), PID control, and PID with meal announcements. BB reflects typical human-in-the-loop control strategies, PID reflects a common control strategy used in preliminary fully closed loop AP applications, PID with meal announcements is based on current AP technology, and requires regular human intervention.

3.3.1 Basal-Bolus Baseline

This baseline is designed to mimic human control and is typical of how an individual with T1D currently controls their blood glucose. In this setting, we modify the standard state representation s_t to include a carbohydrate signal and a cooldown signal (explained below), and to remove all non-current measurements $s_t = [b_t, i_t, c_t, cooldown]$. Note that the inclusion of a carbohydrate signal, or meal announcement, places the burden of providing accurate and timely estimates of meals on the person with diabetes. Each virtual patient in the simulator comes with the parameters necessary to calculate optimal basal insulin rate bas, a correction factor CF, and carbohydrate ratio CR. These three parameters, together with a glucose target b_g define a clinician-recommended policy $\pi(s_t) = bas + (c_t > 0) * (\frac{c_t}{CR} + cooldown * \frac{b_t - b_g}{CF})$ where $cooldown$ is 1 if there have been no meals in the past three hours, otherwise it is 0. This ensures that each meal is only corrected for once, otherwise meals close in time could lead to over-correction and hypoglycemia. These three parameters can be estimated by endocrinologists using previous glucose and insulin information (Walsh et al., 2011). The parameters for our virtual patient population are set according to Xie (2018).

3.3.2 PID Baseline

Variants of PID controllers are already used in commercial AP applications (Garg et al., 2017). A PID controller operates by setting the control variable, here a_t, to the weighted combination of three terms $a_t = k_P P(b_t) + k_I I(b_t) + k_D D(b_t)$ such that the process variable b_t (where t is again the time index) remains close to a specified setpoint b_g. The terms are calculated as follows: i) the proportional term $P(b_t) = \max(0, b_t - b_g)$ increases the control variable proportionally to the distance from the setpoint, ii) the integral term $I(b_t) = \sum_{j=0}^{t}(b_j - b_g)$ acts to correct long-term deviations from the setpoint, and iii) the derivative term $D(b_t) = |b_t - b_{t-1}|$ acts to control a basic estimate of the future, here approximated by the rate of change. The set point and the weights (also called gains) k_P, k_D, k_I are hyperparameters. To compare to the strongest PID controller possible, we tuned these hyperparameters extensively using multiple iterations of grid-search with exponential refinement per-patient. Our final parameters are presented in **Appendix A.2**

Table 1: Average risk, and percent of time Eu/Hypo/Hyperglycemic over 10 days of simulation, 3 runs each for 30 patients (± standard deviation). Hybrid and Non-closed loop approaches (requiring meal announcements) are indicated with *. The approach with the best average score is underlined, the best approach that does not require meal announcements is bolded. Among the approaches that do not require meal announcements, SAC-GRU-Trans achieves the lowest risk and most time Euglycemic.

	Risk	Euglycemia (%)	Hypoglycemia (%)	Hyperglycemia (%)
BB*	21.37 ± 70.70	53.81 ± 12.41	3.15 ± 11.27	43.04 ± 11.39
PID	9.10 ± 6.14	71.03 ± 11.30	**2.29 ± 3.52**	26.67 ± 11.07
PID-MA*	6.15 ± 3.57	75.78 ± 14.48	7.32 ± 8.18	16.89 ± 10.47
SAC-Oracle*	3.21 ± 2.03	86.52 ± 9.40	1.42 ± 2.11	12.07 ± 8.32
SAC-GRU	16.77 ± 54.09	71.43 ± 17.17	9.66 ± 10.16	18.91 ± 14.17
SAC-GRU-Trans	**6.14 ± 2.86**	**75.04 ± 11.12**	6.80 ± 4.55	**18.15 ± 9.14**

PID with Meal Announcements. This baseline, which is designed to be similar to commercially available hybrid closed loop systems (Garg et al., 2017; Ruiz et al., 2012), combines the BB with the PID algorithm into a control algorithm which we call PID with meal announcements (PID-MA). During meals, insulin boluses are calculated and applied as in the BB approach, but instead of using a predetermined fixed basal insulin rate, the PID algorithm controls the basal rate, allowing for adaptation between meals. We similarly tune the gain parameters for the PID algorithm using sequential grid search with exponential refinement.

3.4 Experimental Setup & Evaluation

To measure the utility of deep RL for the task of blood glucose control, we learned policies using the approaches described above, and tested these policies on simulated data with different random seeds across 30 different individuals.

We trained our models (from scratch) for 300 epochs (batch size 256, epoch length 20 days) with an experience replay buffer of size 1e6 and a discount factor of 0.99. We trained our RL models using automatic entropy tuning and sampling actions for exploration (Haarnoja et al., 2018). We optimized the Q, V and π losses using Adam with a learning rate of 10^{-3}. All network hyperparameters were optimized on training seeds on a subset of the virtual patients. Our networks were initialized using PyTorch defaults. When fine-tuning models transferred across patients we then train for 50 epochs with a learning rate of 10^{-4}. All of our code will be made publicly available to allow for replication. For the purpose of anonymous peer-review, we have made our code available on an anonymous google drive account [1].

We measured the performance (average risk) of the policy networks on 10 days of validation data after each epoch. After training, we selected the model that performed the best on these validation runs for testing, also on 10 continuous days of data. We evaluated potential control algorithms using average risk, time spent euglycemic, and time hypo/hyperglycemic. The random seeds controlling noise and meals in the environment were different between training, validation, and test runs. We ran the pipeline three times for each virtual patient.

4 Experiments and Results

We investigate the performance of several different classes of policies under different settings. We compare the performance of the BB controller, the PID with and without meal announcements, and the SAC approaches with the Oracle and learned representation across the thirty virtual patients. In follow-up experiments, we demonstrate the efficiency of transferring learned policies across patients relative to training from scratch, and examine the ability of the RL approach to leverage latent behavioral patterns.

Baseline Models vs. SAC. Results comparing the BB, PID, and PID-MA baselines to the SAC-GRU-Trans network are given in **Figure 2**. Each point represents a different policy, resulting from a different initialization. Despite the variation across individuals, a clear tread emerges: closed-loop control algorithms that can deliver frequent small doses of insulin can significantly outperform a BB

[1] `https://tinyurl.com/y6e2m68b`

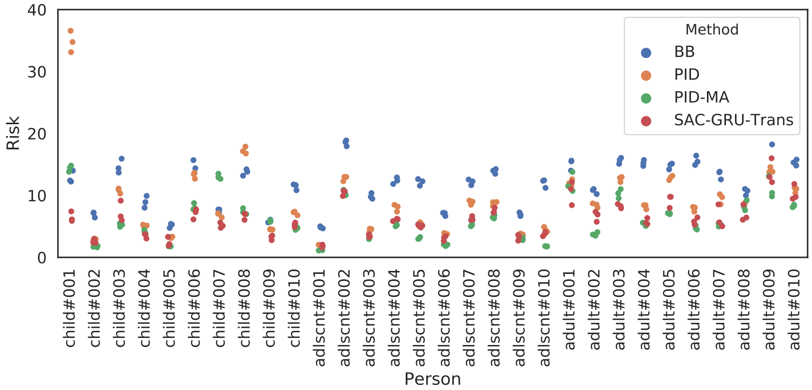


Figure 2: The average risk over 10 days from different methods applied to different simulated patients. Each point corresponds to a different random seed, that controls initialization, the meal schedule, and randomness in training. On average, the SAC and PID-MA methods perform best.

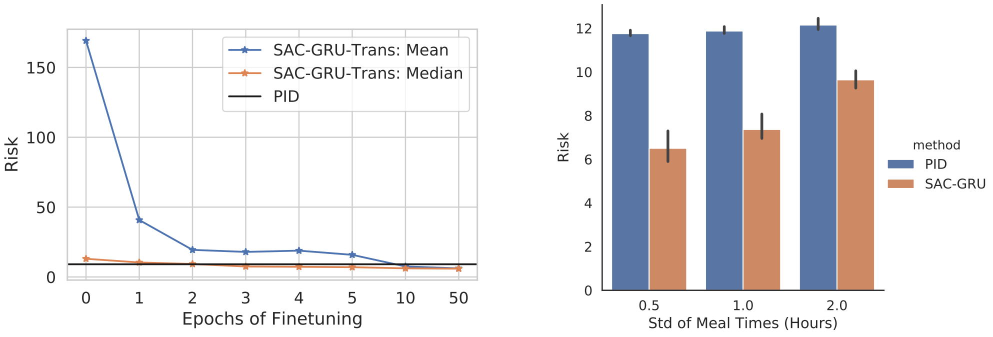
controller. This suggests that, in addition to relieving decision burden, AP systems could lead to overall better blood glucose control. The SAC-GRU-Trans achieved a significantly lower risk than the pure PID (using an independent t-test, $p < 10^{-4}$), and matched the performance of the PID-MA algorithm without requiring meal announcements.

The average risk for many individuals is above the risk threshold for hyper/hypoglycemia of 7.75. This is far from the optimal level of control. However, it is not the case that all time is spent hypo/hyperglycemic. Across patients, approximately 60-80% of time is spent euglycemic, compared with $52\% \pm 19.6\%$ observed in real human control (AyanoTakahara et al., 2015). If insulin is not given well in advance of meals, glucose can increase significantly for a brief period of time, leading to elevated average/mean risk. This skews the distribution of risk towards hyperglycemia and therefore increased risk.

We examine additional models and metrics in the results presented in **Table 1**. We observe that the SAC-Oracle approach is the best across all metrics. This demonstrates an advantage of RL-based control schemes, when given additional information it is simple to improve performance. Among more realistic approaches, PID-MA and SAC-GRU-Trans are comparable in terms of performance. Interestingly, SAC-GRU performs worse on average compared to SAC-GRU-Trans. This is due to occasional catastrophic errors in the policy trained from scratch, where final performance is dangerously poor (5 runs across 3 patients resulted glucose traces with a mean risk of more than 25). The process of transferring and fine-tuning policies eliminates these all of these failures.

Efficient Policy Transfer. While SAC-GRU-Trans achieves stronger performance than SAC-GRU with less patient-specific data, it still requires a large amount for any one individual in a non-simulation setting. In **Figure 3a**, we show the average policy performance by epoch of target training. We see that, in the median case, far less training is required to achieve good performance. For half the individuals, we outperform the PID controller within 3 epochs of fine-tuning (or 60 days). However, without a significant number of update epochs, the learned policies may still result in catastrophic failures which lowers mean performance. With our current approach approximately 10 epochs of updating are required to eliminate these catastrophic failures.

Ability to Adapt to Meals. We hypothesize that one of the potential advantages of RL is its ability to exploit underlying behavioral patterns. To investigate this potential benefit, we explored changing the meal schedule generation procedure outlined in **Algorithm 1** for Adult 1. We removed the ‘snack’ meals (those with occurrence probabilities of 0.3) and set all meal occurrence probabilities to 1 and meal amount standard deviations to 0 (*i.e.*, each day Adult 1 consumes an identical set of meals). We then evaluated both the PID model and the SAC-GRU model on 3 variations of this environment, characterized by the standard deviation of the meal times (either 0.5, 1, or 2 hours). This tests the ability of each method to take advantage of latent patterns in the environment. The results are presented in **Figure 3b**. We observe that, while SAC-GRU achieves lower risk than PID under all



(a)

(b)

Figure 3: a) The impact of fine-tuning SAC-GRU-Trans; performance reported across all patients. While median performance rapidly surpasses the PID (within 3 epochs of fine-tuning), it takes 10 epochs for mean performance to surpass the PID due to catastrophic failures after initial transfer. b) Average risk over 10 days for Adult 1 using different meal schedules. As meal times become more predictable (lower standard deviation), SAC-GRU performs better.

settings, the difference becomes more pronounced as the standard deviation of meal times becomes smaller (and thus meal timing becomes more predictable). This demonstrates that SAC-GRU is better able to leverage latent meal patterns.

5 Discussion and Conclusion

In this work, we develop and explore deep RL algorithms to learn automated blood glucose control policies. When given information about the ground truth state, a soft actor-critic (SAC-Oracle) convincingly outperformed baseline approaches. Without access to the ground-truth state, or even meal announcements, a recurrent SAC outperformed both the BB and PID baselines, matching the performance of a PID with meal announcements. Moreover, this approach was able to significantly improve performance in the presence of a predictable meal schedule.

The use of policy transfer was found to be important in stabilizing performance for the SAC-GRU. Beyond the performance of the learned policies, across our experiments, we found that thousands of days of simulation data were required when training our deep approaches from scratch. However, by transferring policies across individuals and fine-tuning, we were able to learn with far less data (and indeed, such transfer performs better on average than training from scratch).

While these results are encouraging, there are several limitations. First, our results are based on simulation. While the simulator in question is a highly credible one, it may not adequately capture variation across patients or changes in the glucoregulatory system over time. However, as an FDA-approved substitute for animal trials (Kovatchev et al., 2009), success in this simulator is a nontrivial accomplishment. Second, we define a reward function based on risk. Though optimizing this risk function should lead to tight glucose control, it could lead to excess insulin utilization (as its use is unpenalized). Future work could consider resource-aware variants of this reward. Finally, we emphasize that blood glucose control is a safety-critical application. An incorrect dose of insulin could lead to life-threatening situations. Importantly, the proposed approach, though promising, is not ready for deployment. As shown by the worst-case performance of the SAC-GRU method in **Table 1**, deep approaches can fail catastrophically. Going forward, there are several approaches that could be investigated to guarantee acceptable worst-case performance. Using the notion of ‘shielding’ from (Alshiekh et al., 2018), hard limits on insulin informed by blood glucose levels could prevent catastrophic hypoglycemia. Though this, in turn, could limit controller effectiveness in response to rapidly increasing glucose levels. Additionally, approaches that incrementally modify existing safe policies can limit worst-case performance and lead to safer control (Berkenkamp et al., 2017). Despite these limitations, our results clearly demonstrate that deep RL is a promising approach for learning truly closed-loop algorithms for blood glucose control.

References

Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Shiho AyanoTakahara, Kaori Ikeda, Shimpei Fujimoto, Kanae Asai, Yasuo Oguri, Shin-ichi Harashima, Hidemi Tsuji, Kenichiro Shide, and Nobuya Inagaki. Carbohydrate intake is associated with time spent in the euglycemic range in patients with type 1 diabetes. *Journal of Diabetes Investigation*, 6(6):678–686, 2015. ISSN 2040-1124. doi: 10.1111/jdi.12360. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/jdi.12360.

B. Wayne Bequette. A Critical Assessment of Algorithms and Challenges in the Development of a Closed-Loop Artificial Pancreas. *Diabetes Technology & Therapeutics*, 7(1):28–47, February 2005. ISSN 1520-9156, 1557-8593. doi: 10.1089/dia.2005.7.28. URL http://www.liebertpub.com/doi/10.1089/dia.2005.7.28.

Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *Advances in neural information processing systems*, pp. 908–918, 2017.

Melanie K Bothe, Luke Dickens, Katrin Reichel, Arn Tellmann, Bjrn Ellger, Martin Westphal, and Ahmed A Faisal. The use of reinforcement learning algorithms to meet the challenges of an artificial pancreas. *Expert Review of Medical Devices*, 10(5):661–673, September 2013. ISSN 1743-4440, 1745-2422. doi: 10.1586/17434440.2013.827515. URL http://www.tandfonline.com/doi/full/10.1586/17434440.2013.827515.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *EMNLP*, June 2014. URL http://arxiv.org/abs/1406.1078. arXiv: 1406.1078.

William Clarke and Boris Kovatchev. Statistical tools to analyze continuous glucose monitor data. *Diabetes Technology & Therapeutics*, 11, 2009. ISSN 1520-9156, 1557-8593. doi: 10.1089/dia.2008.0138. URL http://www.liebertpub.com/doi/10.1089/dia.2008.0138.

Ignasi Clavera, Anusha Nagabandi, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt: Meta-learning for model-based control. *arXiv preprint arXiv:1803.11347*, 3, 2018.

Claudio Cobelli, G Federspil, G Pacini, A Salvan, and C Scandellari. An integrated mathematical model of the dynamics of blood glucose and its hormonal control. *Mathematical Biosciences*, 58 (1):27–60, 1982.

Ronald D Coffen and Lynnda M Dahlquist. Magnitude of type 1 diabetes self-management in youth health care needs diabetes educators. *The Diabetes Educator*, 35(2):302–308, 2009.

Mariano De Paula, Gerardo G. Acosta, and Ernesto C. Martnez. On-line policy learning and adaptation for real-time personalization of an artificial pancreas. *Expert Syst. Appl.*, 42(4):2234–2255, 2015. ISSN 0957-4174. doi: 10.1016/j.eswa.2014.10.038. URL http://dx.doi.org/10.1016/j.eswa.2014.10.038.

Diabetes Control and Complications Trial Research Group. Resource utilization and costs of care in the diabetes control and complications trial. *Diabetes Care*, 18(11):1468–1478, 1995.

Ian Fox, Lynn Ang, Mamta Jaiswal, Rodica Pop-Busui, and Jenna Wiens. Deep multi-output forecasting: Learning to accurately predict blood glucose trajectories. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pp. 1387–1395. ACM, 2018. ISBN 978-1-4503-5552-0. doi: 10.1145/3219819.3220102. URL http://doi.acm.org/10.1145/3219819.3220102.

Satish K. Garg, Stuart A. Weinzimer, William V. Tamborlane, Bruce A. Buckingham, Bruce W. Bode, Timothy S. Bailey, Ronald L. Brazg, Jacob Ilany, Robert H. Slover, Stacey M. Anderson, Richard M. Bergenstal, Benyamin Grosman, Anirban Roy, Toni L. Cordero, John Shin, Scott W. Lee, and Francine R. Kaufman. Glucose Outcomes with the In-Home Use of a Hybrid Closed-Loop Insulin Delivery System in Adolescents and Adults with Type 1 Diabetes. *Diabetes Technology & Therapeutics*, 19(3):155–163, January 2017. ISSN 1520-9156. doi: 10.1089/dia.2016.0421. URL https://www.liebertpub.com/doi/full/10.1089/dia.2016.0421.

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.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *International Conference on Machine Learning*, pp. 1861–1870, July 2018. URL http://proceedings.mlr.press/v80/haarnoja18b.html.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

A. H. Kadish. Automation Control of Blood Sugar. I. a Servomechanism for Glucose Monitoring and Control. *The American journal of medical electronics*, 3:82–86, 1964.

Marie E Kerl. Diabetic ketoacidosis: pathophysiology and clinical and laboratory presentation. *Compendium*, 23(3):220–228, 2001.

Predrag Klasnja, Shawna Smith, Nicholas J. Seewald, Andy Lee, Kelly Hall, Brook Luers, Eric B. Hekler, and Susan A. Murphy. Efficacy of contextually tailored suggestions for physical activity: A micro-randomized optimization trial of HeartSteps. *Annals of Behavioral Medicine*, 53(6): 573–582, 2019. ISSN 0883-6612. doi: 10.1093/abm/kay067. URL https://academic.oup.com/abm/article/53/6/573/5091257.

Matthieu Komorowski, Leo A. Celi, Omar Badawi, Anthony C. Gordon, and A. Aldo Faisal. The Artificial Intelligence Clinician learns optimal treatment strategies for sepsis in intensive care. *Nature Medicine*, pp. 1, 2018.

Boris P Kovatchev, Marc Breton, Chiara Dalla Man, and Claudio Cobelli. In silico preclinical trials: A proof of concept in closed-loop control of type 1 diabetes. *Journal of Diabetes Science and Technology*, 3(1):44–55, 2009.

Akio Kuroda, Hideaki Kaneto, Tetsuyuki Yasuda, Munehide Matsuhisa, Kazuyuki Miyashita, Noritaka Fujiki, Keiko Fujisawa, Tsunehiko Yamamoto, Mitsuyoshi Takahara, Fumie Sakamoto, Taka-aki Matsuoka, and Iichiro Shimomura. Basal insulin requirement is 30% of the total daily insulin dose in type 1 diabetic patients who use the insulin pump. *Diabetes Care*, 34(5):1089–1090, 2011. ISSN 0149-5992, 1935-5548. doi: 10.2337/dc10-2149. URL https://care.diabetesjournals.org/content/34/5/1089.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. ISSN 0028-0836. doi: 10.1038/nature14236. URL http://www.nature.com/nature/journal/v518/n7540/abs/nature14236.html.

Jordan E. Pinsker, Joon Bok Lee, Eyal Dassau, Dale E. Seborg, Paige K. Bradley, Ravi Gondhalekar, Wendy C. Bevier, Lauren Huyett, Howard C. Zisser, and Francis J. Doyle. Randomized Crossover Comparison of Personalized MPC and PID Control Algorithms for the Artificial Pancreas. *Diabetes Care*, pp. dc152344, June 2016. ISSN 0149-5992, 1935-5548. doi: 10.2337/dc15-2344. URL http://care.diabetesjournals.org/content/early/2016/06/10/dc15-2344.

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. *arXiv preprint arXiv:1704.06300*, 2017.

Jessica L. Ruiz, Jennifer L. Sherr, Eda Cengiz, Lori Carria, Anirban Roy, Gayane Voskanyan, William V. Tamborlane, and Stuart A. Weinzimer. Effect of Insulin Feedback on Closed-Loop Glucose Control: A Crossover Study. *Journal of Diabetes Science and Technology*, 6(5):1123–1130, September 2012. ISSN 1932-2968. doi: 10.1177/193229681200600517. URL https://doi.org/10.1177/193229681200600517.

Sara Trevitt, Sue Simpson, and Annette Wood. Artificial pancreas device systems for the closed-loop control of type 1 diabetes. *Journal of Diabetes Science and Technology*, 10(3):714–723, 2015. ISSN 1932-2968. doi: 10.1177/1932296815617968. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5038530/.

Jaakko Tuomilehto. The emerging global epidemic of type 1 diabetes. *Current diabetes reports*, 13 (6):795–804, 2013.

Roberto Visentin, Chiara Dalla Man, Boris Kovatchev, and Claudio Cobelli. The university of virginia/padova type 1 diabetes simulator matches the glucose traces of a clinical trial. *Diabetes technology & therapeutics*, 16(7):428–434, 2014.

John Walsh, Ruth Roberts, and Timothy Bailey. Guidelines for optimal bolus calculator settings in adults. *Journal of Diabetes Science and Technology*, 5(1):129–135, 2011. ISSN 1932-2968. doi: 10.1177/193229681100500118. URL https://doi.org/10.1177/193229681100500118.

Wei-Hung Weng, Mingwu Gao, Ze He, Susu Yan, and Peter Szolovits. Representation and reinforcement learning for personalized glycemic control in septic patients. *NeurIPS 2017 ML4H Workshop*, 2017. URL http://arxiv.org/abs/1712.00654.

Jinyu Xie. Simglucose, 2018. URL https://github.com/jxx123/simglucose.

Wen-Peng You and Maciej Henneberg. Type 1 diabetes prevalence increasing globally and regionally: the role of natural selection and life expectancy at birth. *BMJ Open Diabetes Research and Care*, 4(1):e000161, 2016.

Taiyu Zhu, Kezhi Li, Pau Herrero, Jianwei Chen, and Pantelis Georgiou. A deep learning algorithm for personalized blood glucose prediction. *IJCAI Knowledge Discovery in Healthcare Data Workshop*, 2018.

A Appendix

A.1 Meal Generation Algorithm

Algorithm 1 Generate Meal Schedule

```
**Input:** body weight $w$, number of days $n$
$MealOcc = [0.95, 0.3, 0.95, 0.3, 0.95, 0.3]$
$TimeLowerBounds = [5, 9, 10, 14, 16, 20] * 12$
$TimeUpperBounds = [9, 10, 14, 16, 20, 23] * 12$
$TimeMean = [7, 9.5, 12, 15, 18, 21.5] * 12$
$TimeStd = [1, .5, 1, .5, 1, .5] * 12$
$AmountMean = [0.7, 0.15, 1.1, 0.15, 1.25, 0.15] * w$
$AmountStd = AmountMean * 0.15$
$Days = []$
**for** $i \in [1, \dots, n]$ **do**
  $M = [0]_{j=1}^{288}$
  **for** $j \in [1, \dots, 6]$ **do**
    $m \sim Binomial(MealOcc[j])$
    $lb = TimeLowerBounds[j]$
    $ub = TimeUpperBounds[j]$
    $\mu_t = TimeMean[j]$
    $\sigma_t = TimeStd[j]$
    $\mu_a = AmountMean[j]$
    $\sigma_a = AmountStd[j]$
    **if** $m$ **then**
      $t \sim Round(TruncNormal(\mu_t, \sigma_t, lb, ub))$
      $c \sim Round(max(0, Normal(\mu_a, \sigma_a)))$
      $M[t] = c$
    **end if**
  **end for**
  $Days.append(M)$
**end for**
```

A.2 PID AND PID-MA PARAMETERS

	k_p	k_i	k_d
child#001	-1.00E-05	-3.68E-08	-7.59E-04
child#002	-3.49E-05	-3.49E-07	-3.98E-03
child#003	-6.31E-05	-2.23E-08	-1.00E-03
child#004	-3.49E-05	-3.49E-07	-1.00E-03
child#005	-1.00E-04	-6.31E-07	-2.87E-03
child#006	-6.31E-05	-2.87E-08	-1.00E-03
child#007	-1.00E-05	-3.49E-07	-2.51E-03
child#008	-1.93E-08	-4.72E-08	-1.00E-03
child#009	-1.00E-05	-3.98E-07	-1.00E-03
child#010	-4.98E-07	-3.49E-07	-2.09E-03
adolescent#001	-2.87E-06	-1.00E-06	-1.00E-02
adolescent#002	-5.53E-09	-4.54E-12	-1.00E-02
adolescent#003	-1.00E-04	-3.49E-07	-3.98E-03
adolescent#004	-6.74E-08	-6.74E-10	-1.00E-02
adolescent#005	-4.54E-10	-2.87E-08	-1.00E-02
adolescent#006	-1.93E-08	-3.49E-06	-6.31E-03
adolescent#007	-1.07E-07	-1.00E-07	-6.31E-03
adolescent#008	-6.74E-08	-8.21E-09	-1.00E-02
adolescent#009	-2.35E-07	-1.00E-06	-3.98E-03
adolescent#010	-1.58E-09	-1.00E-07	-1.00E-02
adult#001	-8.32E-05	-1.00E-07	-1.00E-02
adult#002	-3.02E-04	-1.00E-07	-1.00E-02
adult#003	-2.87E-06	-6.07E-08	-1.00E-02
adult#004	-2.87E-05	-3.49E-07	-3.98E-03
adult#005	-1.00E-04	-1.00E-07	-1.00E-02
adult#006	-1.00E-04	-5.75E-07	-1.00E-02
adult#007	-1.35E-06	-1.58E-07	-1.00E-02
adult#008	-4.72E-06	-1.00E-07	-1.00E-02
adult#009	-1.00E-04	-1.00E-07	-1.00E-02
adult#010	-6.31E-05	-1.00E-07	-1.00E-02

Table 2: PID parameters

	k_p	k_i	k_d
child#001	-3.63E-05	-3.98E-07	-6.31E-04
child#002	-3.98E-05	-1.00E-06	-2.51E-03
child#003	-4.72E-06	-2.87E-08	-1.00E-03
child#004	-6.31E-05	-1.00E-06	-1.58E-03
child#005	-1.58E-04	-4.79E-07	-2.51E-03
child#006	-3.49E-05	-1.00E-07	-1.00E-03
child#007	-2.29E-04	-3.88E-09	-2.51E-03
child#008	-1.00E-05	-1.00E-07	-1.00E-03
child#009	-2.51E-05	-4.37E-07	-4.37E-04
child#010	-6.39E-07	-3.98E-07	-1.00E-03
adolescent#001	-4.54E-10	-4.54E-12	-1.00E-02
adolescent#002	-4.54E-10	-1.00E-07	-1.00E-03
adolescent#003	-1.91E-05	-5.75E-07	-1.74E-03
adolescent#004	-2.23E-06	-6.31E-07	-2.51E-03
adolescent#005	-4.54E-10	-1.00E-06	-3.02E-03
adolescent#006	-4.54E-10	-1.00E-05	-2.51E-03
adolescent#007	-4.54E-10	-4.79E-07	-3.49E-03
adolescent#008	-4.54E-10	-1.00E-07	-1.00E-03
adolescent#009	-4.54E-10	-6.92E-07	-1.58E-03
adolescent#010	-6.31E-05	-6.31E-07	-6.31E-03
adult#001	-1.00E-05	-1.00E-07	-3.49E-03
adult#002	-4.54E-10	-6.31E-07	-6.31E-03
adult#003	-4.54E-10	-4.37E-07	-1.00E-03
adult#004	-1.74E-06	-6.31E-07	-1.00E-03
adult#005	-4.98E-07	-1.00E-07	-1.00E-03
adult#006	-4.54E-10	-1.00E-06	-2.87E-03
adult#007	-3.73E-07	-6.92E-07	-2.75E-03
adult#008	-1.00E-04	-3.49E-07	-3.49E-03
adult#009	-4.54E-10	-1.00E-07	-3.49E-03
adult#010	-4.54E-10	-1.00E-07	-1.00E-03

Table 3: PID-MA parameters