Offline RL for T1 Diabetes Management

Aman Chopra
Columbia University
ac5140@columbia.edu

Priyanka Balakumar
Columbia University
pb2893@columbia.edu

Akshit Bhalla
Columbia University
ab5661@columbia.edu

Martin Roux
Columbia University
mr4396@columbia.edu

Abstract

The growth of deep neural networks has facilitated breakthroughs in medical advancements and have allowed us to deploy powerful algorithms on wearable devices.This paper discusses a reinforcement learning agent for type 1 diabetes management that can act as an artificial pancreas. It was trained on data emerging from a continuous glucose monitor (CGM), a wearable Apple Watch, and an insulin pump. The agent learns a sequence of blood glucose, heart rate, and insulin on board levels and using a behavioral policy, takes actions and transitions to a new state in an offline setting. A Deep Q Network and Policy Gradient were used for the learning process. While legal restrictions inhibit the actual deployment of the algorithm, it was successfully tested with observed data and found to be robust.

1 Introduction

Current technology in T1 Diabetes management has witnessed significant advancements with the development of insulin pumps and CGMs. Together, they allow for precise and convenient insulin dosing, contributing to better glycemic control. However, the integration of data from wearables, such as fitness trackers and smartwatches, presents a new frontier in diabetes management. These devices capture additional contextual data, including attributes surrounding physical activity and sleep patterns. Offline RL is a particularly promising avenue, allowing models to learn from historical data even in the absence of real-time connectivity. This project aims to construct a reinforcement learning model utilizing data extracted from wearables and incorporating this information into RL models to refine T1D management strategies.

2 Related Work

Over the last few years, research in the area of RL applications of diabetes has been growing. Tejedor et al (5) have conducted the most comprehensive literature review in this space. Most research was conducted and published between 2012 and 2019. We can view a summary in Table 1.

Overall, actor-critic learning and Q-learning with approximate solution methods were most popular. Most state spaces were continuous with glucose level as the only feature. For the action space, most implementations use a continuous action space with only insulin dosages.

3 Data

All data utilized in this project originates from a real-world data collected over the past 6 months. While the dataset is extensive, our focus narrows down to four attributes: heart rate (HR), insulin on board (IOB), blood glucose levels, and insulin dosage. IOB was computed by assuming that active insulin is linearly absorbed by the body over a 4-hour span. The data is time-stamped at varying intervals, from seconds (HR data) to 5 minutes (CGM data) to hours (insulin dosage data). To create consistent sampling intervals, we rounded all timestamps to the nearest 5 minutes, and aggregated each feature. BG and HR were aggregated by mean, insulin dosage by sum, and IOB by sum. Prior to building the transition model, we conducted EDA:

- 45,762 rows spanning dates from 2/18/23 - 8/19/23.
- Blood glucose readings are mostly normal, centered at 144 mg/dL with a std. dev. of 59 mg/dL.
- HR readings are mostly normal, centered at 66 bpm with a std. dev. of 19 bpm.
- Insulin dosage amounts are severely right skewed with a mean of 0.05 units, median of 0 units, and std. dev. of 0.38 units. This

Table 1: Literature Review of Papers

Ref.	Subjects	Type of Diabetes	Data Source	Class of RL	Exploitation vs Exploration	State Space	Action Space	Planning
(6)	1 in silico patient.	T1DM	AIDA model (7)	GPRL	BAL	CONT. BG and insulin	CONT. insulin	Model based
(8)	52 real patients	T2DM	Clinical data. Clinical study.	Learning automaton	Gaussian distribution.	CONT. BG	CONT. insulin	Model free
(9)	28 in silico patients	T1DM	UVA / PADOVA Simulator (10)	AC	N / A	CONT. BG	CONT. insulin	Model free
(11)	70 real patients.	N / A	Clinical data. Public data set.	Q - learning	N / A	DISC. BG	DISC. insulin	Model free
(12)	3 in silico patients	T1DM	Bergman's minimal model (13)	Q - learning	E - greedy policy	DISC. BG	DISC. insulin	Model free

makes sense, as insulin delivery amounts are 0 at most timestamps.

- IOB amounts are right skewed with a mean of 1.2 units, median of 0.54 units, and std. dev. of 1.6 units.

4 Experiments

4.1 Defining a Markov Decision Process

An MDP is defined as a tuple (S, A, P, R, γ). S contains all the states s_i, $0 <= i < |S|$ that exist in the state space. A contains all the actions a_j, $0 <= j < |A|$ that exist in the action space. P represents the transition model $P(s_x, a_y, s_z)$, $0 <= x, y < |S|$, $0 <= y < |A|$ that indicates the probability of transitioning to state s_z, given that the agent is in state s_x and takes action a_y. R is the reward model $R(s_x, a_y, s_z)$ indicating the reward obtained from making the transition from s_x to s_z through a_y. Lastly, γ represents the discount factor which ranges from 0 to 1. Solving the MDP involves creating an optimal policy π^* which maximizes the total reward, defined by $\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$ (15).

4.1.1 State Space Representation

We represent a state using the data we gathered from Apple Health, Dexcom CGM, and Tandem Tslim Insulin Pump. We use 3 features in our state space: BG blood glucose, IOB insulin on board, and HR heart rate. Each feature includes 18 lags of data; since data was aggregated at 5-minute intervals, this equates to 90 minutes of data in a single state. An example of a state representation would be $s_1 = (BG_1, ..., BG_{18}, HR_1, ..., HR_{18}, IOB_1, ...IOB_{18})$. The number of lags is a hyperparameter we can tune, but we selected this initially as research shows that it takes roughly 90 minutes for insulin to peak and affect blood sugar levels (1).

4.1.2 Action Space Representation

We use 1 feature in our action space: I, units of insulin delivered. The feature is continuous in the original data, but we truncated the decimals, to make the action space discrete. Research shows that basic Q-learning could diverge with continuous action spaces (4), so we truncated the decimals in I to make the values discrete. Because our methods involved using more traditional forms of RL, deep Q-learning and policy gradient, rather than soft-actor critic methods, we chose to keep our action space simple.

4.1.3 Reward

We considered 2 reward functions: a simple reward and the Magni (2) reward.

The simple reward is defined as follows:

$$SR(BG_t) = \begin{cases} 1 & BG_t \in [90, 130] \\ 0 & BG_t \in [50, 90) \vee (130, 250] \\ -100 & \text{otherwise} \end{cases}$$

This reward was designed based on research that suggests that a simple function encourages longer episodes. More specifically, (3) studied using a continuous complex reward function but noticed that the agent becomes aggressive with injecting more insulin to avoid continuous negative rewards. However, these large amounts of insulin will eventually cause hypoglycemia.

The Magni reward is defined as the negative of the Magni risk function:

$$MR(BG_t) = 10 * (3.5506 * ((ln(BG_t))^{0.8353} - 3.7932))^2 \quad (1)$$

Across most papers, this reward function resulted in the best performance (5).

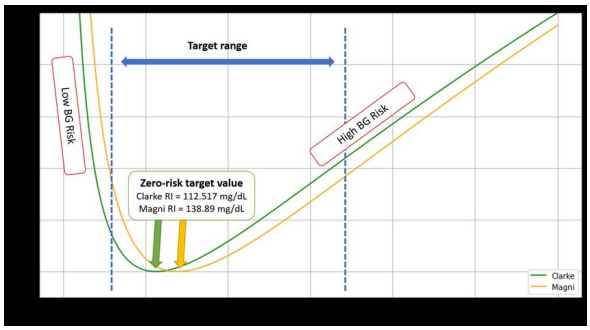

Figure 1: Magni Risk Function

4.2 Algorithms

4.2.1 Q-Learning

Q-learning involves estimating Q^*, the optimal state-action-value function, which results in the optimal policy $\pi^* = argmax_{a \in A} Q(s_i, a)$. The Q-values are updated during each iteration of training using an update rule. We leveraged the temporal-difference (TD) update rule: $Q(s_t, a_t) = (1-\alpha) * Q(s_t, a_t) + \alpha (r_t + \max_a \gamma * Q(s_{t+1}, a))$ (15). Because our state space is continuous in nature, we use deep Q-learning (DQN) rather than tabular Q-learning. Our neural network consisted of 3 layers with the Relu activations and a final layer 10 outputs (dimensionality of action space) with no activation.

We maintained a replay buffer R to store transition tuples $(s_t, a_t, r_t, s_{t+1}$ to minimize the Bellman error: $(Q_\theta(s_t, a_t) - E[r_t + \gamma \max_a Q_\theta(s_{t+1}, a)])^2$. R allows us to store experiences during exploration and sample batches of experiences during training. We also maintain a target network, updated every τ timestamps, to compute the targets in the bellman error described above to stabilize learning.

4.2.2 Policy Gradient

Policy gradient is another approach to RL that involves finding the optimal policy π^* directly, rather than finding the optimal value function V^* or state-action value function Q^* and then extracting the policy. The goal is to maximize the expected value of a policy. The expected value of a policy π with parameters p is defined as $J(p) = V^{\pi_p}(s_0)$ where s_0 is the starting state (4). We then apply gradient ascent to solve for the policy's parameters. Like DQN, because our state space is continuous in nature, we used function approximation rather than tables to represent the policy. Our neural network consisted of 3 layers with ReLu activations and a final layer with a Softmax activation over all possible actions.

We initially estimated the state-action-value function Q using Monte Carlo estimates but then transitioned to using the same network defined in DQN to help stabilize learning.

For both DQN and policy gradient, the input dimension is 54 as each of the 3 features (BG, HR, IOB have 18 lags each) and the output dimension is 10 as there are 10 discrete values for I, the action representing the amount of insulin delivered.

4.2.3 Exploration-Exploitation

In the ideal scenario, we would have first explored with an ϵ-greedy strategy to balance exploration and exploitation. When trying this, we realized that the agent could only take actions based on the behavioral (data-based) policy, because our transition model is represented by a lookup table, as shown in Figure 2.

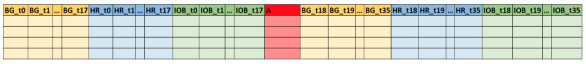

Figure 2: Transition Model Mapping Current State and Action to Next State

This means that if the agent is in a current state s_i, defined by row i of the lookup table, it can only take action a_i to obtain a next state. If the agent, took a different action a_j, the environment wouldn't be able to return a next state, since a_j was never taken in the data for s_i. Choosing actions using the lookup table is very restrictive and prevents the agent from exploring or exploiting. A solution to this would be to build a regression model mapping a current state and action to a next state. Another solution would be to discretize or bin the state space to create more options for taking different actions given a current state. We were able to build a simple regression model with 2 dense layers with Relu activations and a final output layer with no activation (54 dimensions, as this is the size of the state space). We used mean-square error as the loss. This model will allow us to follow an effective ϵ-greedy strategy.

5 Results

5.0.1 Evaluation Metrics

The evaluation of the policy's performance relies on essential metrics centered around cumulative rewards during training episodes. The training reward represents the total accumulated reward in the

most recent training episode, reflecting the impact of the policy's decisions on blood glucose levels throughout the episode. Complementing this, the training reward moving average is computed over a fixed window of the most recent 100 training episodes. Both metrics, logged using the WandB library, offer a comprehensive assessment of the policy's efficacy in diabetes management. The training reward provides immediate feedback, while the moving average reveals the policy's sustained impact and adaptability over time.

5.0.2 DQN Algorithm Results

In this section, we scrutinize the outcomes of the DQN algorithm Our analysis is conducted under the framework of the hyperparameters previously specified. The observed training reward exhibits a conspicuous pattern of fluctuations. Specifically, when considering a simple reward, the training reward oscillates erratically between 200 and -200, indicating a suboptimal learning pattern.

Figure 3: Training reward, simple reward

Notably, this behavior prompts a closer examination of the training reward's moving average, depicted in the subsequent visualization. Despite the training reward's inherent volatility, the moving average over a 100-step window presents a more subdued and negative trend. Intriguingly, this moving average demonstrates an increasing trend as the maximum number of steps per episode rises. However, this behavior is not consistent, and convergence to a consistently high value is not achieved in all instances. This nuanced analysis sheds light on the algorithm's performance nuances and underscores the importance of further exploration and refinement.

Upon concluding the training phase with the simple reward, our subsequent exploration involved

Figure 4: Training reward moving average, simple reward

training the model on the Magni Reward, aligning more closely with our model's objectives. Despite persistent fluctuations in the training reward, the overall average remains notably low. The visual representation of the moving average training reward underscores this consistency, as the average consistently maintains a low level.

Figure 5: Training reward moving average, Magni reward

Notably, this model introduces a constraint where the agent can only take actions based on a data-driven, behavioral policy. In this constrained setting, the model is limited and it cannot lead to a high reward. In response to this constraint, we opted to construct a regression model that maps a current state and action to a next state. The ensuing results are striking:

This regression model yields a high reward that converges to 0. The agent demonstrates a sophisticated understanding of when to administer insulin

Figure 6: Training reward moving average, Magni reward with regressor

to regulate both blood glucose and heart rate, attesting to the efficacy of the refined model.

5.0.3 Policy Gradient Algorithm Results

In this section, we delve into the outcomes of the Policy Gradient algorithm, focusing on both the training reward and its corresponding moving average. With the application of the specified hyperparameters, our observations unveil a training reward marked by distinctive fluctuations. Notably, these fluctuations exhibit a lower amplitude and lower frequency compared to the DQN algorithm, showcasing a nuanced pattern.

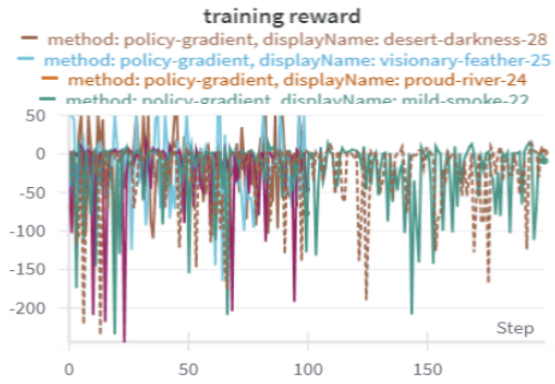

Figure 7: Training reward, simple reward

The observed simple reward manifests with varying characteristics across different results. Plots in green and purple consistently oscillate between 0 and a large negative number, while others in brown and blue exhibit fluctuations from a large positive to a large negative number. Despite the variability in reward behavior, the moving average remains relatively lower than that of the DQN algorithms, fluctuating around -20 as opposed to -30.

However, a notable distinction emerges: the moving average does not exhibit any clear indication of an increase or convergence around a value

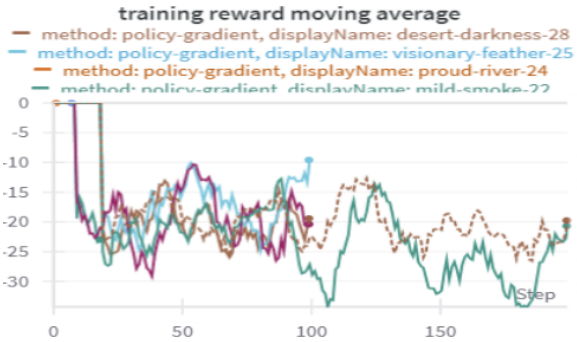

Figure 8: Training reward moving average, simple reward

close to 0. Even with exhaustive exploration, we were unable to identify a set of hyperparameters that induces convergence with the simple reward. This observation highlights a key challenge in achieving convergence with the current configuration.

Following the training of the algorithm using the simple reward, our subsequent exploration involved training it with the Magni Reward, leading to the following outcomes.

Figure 9: Training reward moving average, Magni reward

Notably, the fluctuations are less frequent and less impactful. This shift in behavior is indicative of positive results, as the moving average consistently remains above -5. A lower reward in absolute value implies that the agent successfully maintains blood glucose and heart rate within the desired range. This algorithm showcases promising performance without the need for a regressor to achieve a high reward.

These findings suggest that this algorithm and the DQN algorithm with regressor, particularly when guided by the Magni reward, holds significant promise for our problem. The reduced fluc-

tuations and consistently low moving average indicate a favorable capacity to regulate blood glucose and heart rate without resorting to more complex modeling structures.

6 Conclusion

The agent exhibits evidence of learning under the stated conditions. However, there's still scope for improvement. For example, append more features like sleep data, V02, meal intake, and workout data could provide personalized insights. Experimenting with various reward functions and designing a continuous action space are other ways of revisiting the MDP setup. Furthermore, building a transition model from offline data using function approximation instead of doing table lookup is a reasonable approach. Employing better exploitation-exploration techniques has shown to improve performance in some experiments. Some research points to SAC and PPO techniques that could boost performance with similar data. In the future, processing reduced dimensions by representing lags with moving averages, rather than retaining the full history might be another direction of improvement. Mature evaluation approaches such as testing against virtual patients using FDA-approved UVA/Padova simulator would provide a more comprehensive evaluation. Finally, in the ideal scenario, deploying the agent (policy) on a personal insulin pump would be the ultimate goal.

References

[1] Dierks, M. H. (2021, August 23). Insulin peak times explained [w/ charts]. AgaMatrix. https://agamatrix.com/blog/insulin-peak-times/.

[2] Emerson, H., Guy, M., amp; McConville, R. (2023). Offline reinforcement learning for safer blood glucose control in people with type 1 diabetes. Journal of Biomedical Informatics, 142, 104376. https://doi.org/10.1016/j.jbi.2023.104376.

[3] Viroonluecha, P., Egea-Lopez, E., amp; Santa, J. (2021). Evaluation of Blood Glucose Level Control in Type 1 Diabetic Patients Using Deep Reinforcement Learning. https://doi.org/10.21203/rs.3.rs-1095721/v1.

[4] Policy gradients. Policy gradients - Introduction to Reinforcement Learning. (n.d.). https://gibberblot.github.io/rl-notes/single-agent/policy-gradients.html.

[5] Tejedor, M., Woldaregay, A. Z., amp; Godtliebsen, F. (2020). Reinforcement learning application in diabetes blood glucose control: A systematic review. Artificial Intelligence in Medicine, 104, 101836. https://doi.org/10.1016/j.artmed.2020.101836.

[6] De Paula, M., Acosta, G. G., amp; Martínez, E. C. (2015). On-line policy learning and adaptation for real-time personalization of an artificial pancreas. Expert Systems with Applications, 42(4), 2234–2255. https://doi.org/10.1016/j.eswa.2014.10.038.

[7] Lehmann, E. D., amp; Deutsch, T. (1992). A physiological model of glucose-insulin interaction in type 1 diabetes mellitus. Journal of Biomedical Engineering, 14(3), 235–242. https://doi.org/10.1016/0141-5425(92)90058-s.

[8] Akbari Torkestani, J. and E. Ghanaat Pisheh, A learning automata-based blood glucose regulation mechanism in type 2 diabetes. Control Engineering Practice, 2014. 26: p. 151-159.

[9] Daskalaki, E., Diem, P., amp; Mougiakakou, S. G. (2013). An actor–critic based controller for glucose regulation in type 1 diabetes. Computer Methods and Programs in Biomedicine, 109(2), 116–125. https://doi.org/10.1016/j.cmpb.2012.03.002.

[10] Man, C. D., Micheletto, F., Lv, D., Breton, M., Kovatchev, B., amp; Cobelli, C. (2014). The UVA/padova type 1 diabetes simulator. Journal of Diabetes Science and Technology, 8(1), 26–34. https://doi.org/10.1177/1932296813514502.

[11] Patil, P., Kulkarni, P., amp; Shirsath, R. (2014). Sequential decision making using Q Learning Algorithm for diabetic patients. Advances in Intelligent Systems and Computing, 313–321. https://doi.org/10.1007/978-81-322-2126-5_35.

[12] Yasini, S., M.B. Naghibi-Sistani, and A. Karimpour, Agent-based Simulation for Blood Glucose Control in Diabetic Patients. International Journal of Applied Science, Engineering and Technology, 2009. 5: p. 40-47.

[13] Bergman, R. N. (2005). Minimal model: Perspective from 2005. Hormone Research in Paediatrics, 64(Suppl. 3), 8–15. https://doi.org/10.1159/000089312.

[14] Palumbo, P., Panunzi, S., amp; De Gaetano, A. (2007a). Qualitative behavior of a family of delay-differential models of the glucose-insulin system. Discrete and Continuous Dynamical Systems - B, 7(2), 399–424. https://doi.org/10.3934/dcdsb.2007.7.399.

[15] Sutton, R. S., Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). The MIT Press.