# Data-Driven Solutions for Blood Glucose Management

by

Harry Rubin-Falcone

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2024

Harry Rubin-Falcone

hrf@umich.edu

ORCID iD: 0000-0002-4185-9394

## ACKNOWLEDGEMENTS

First, I extend my deepest gratitude to my advisor, Jenna Wiens, whose relentless drive and remarkable expertise have not only pushed me to exceed my own expectations but also profoundly shaped my academic journey. Without her support and guidance this dissertation would have been impossible. I also offer my heartfelt thanks to Joyce Lee, whose collaboration and insight throughout the degree has been invaluable, and Emily Mower Provost and Nikola Banovic, who have been inspiring committee members.

MLD3 has been a great lab to work with, and I offer my thanks to them all. Especially Ian Fox and Jung Min Lee, who have been my collaborators in blood glucose ML, and Meera Krishnamoorthy, who was my classes buddy.

Thanks to all my friends and family who I love so dearly, especially my wife Hannah, my dog Marvin, and our unborn child.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF APPENDICES

# LIST OF ACRONYMS

|  |  |
|---|---|
| **BG** | Blood Glucose |
| **T1D** | Type 1 Diabetes |
| **CGM** | Continuous Glucose Monitor |
| **BB** | Basal bolus control strategy |
| **ML** | Machine Learning |
| **RNN** | Residual Neural Network |
| **LSTM** | Long Short-term Memory Network |
| **DAE** | Denoising Autoencoder |

# ABSTRACT

Type 1 diabetes (T1D) affects millions of people worldwide. People with T1D must regularly monitor their blood glucose (BG) level and administer insulin to manage it. This process is burdensome, necessitating frequent measurements, meal size estimation, and bolus calculations. Automated management solutions have been proposed, but still require patients to accurately estimate meal sizes and manually update user parameters. We aim to discern and address challenges in the utilization of data-driven approaches for BG management. First, estimating meal size is challenging, resulting in noisy carbohydrate counts, which hamper BG management. We address noise in patient-reported carbohydrate estimates by developing a novel training method for denoising autoencoders. Our approach leverages the relationship between carbohydrates and the BG signal. Second, while data-driven approaches could obviate the need for manual patient updates via their capacity for online learning, current approaches fall short of the level of accuracy required for safe automated BG management. More specifically, learning the impacts of carbohydrates and insulin boluses on future BG values is challenging due to the relative sparsity of these variables and the correlation between them. We propose a forecasting approach that accounts for the relative sparsity of bolus and carbohydrate values by isolating and constraining their effects to align with domain knowledge. In addition, we address bolus and carbohydrate entanglement with an approach that leverages correction bolus values to disentangle individual variable effects. Combined, these contributions represent a clear step towards data-driven BG management, offering the potential to the reduce patient burden associated with T1D.

CHAPTER 1

# Introduction

Type 1 diabetes (T1D) is a global health concern that currently affects over 8 million individuals worldwide, and its prevalence is projected to increase more than 60% by 2040 [Gregory et al., 2022]. Individuals with T1D possess an impaired glucose regulation mechanism. In contrast to healthy individuals whose bodies automatically control blood glucose (BG) levels, individuals with T1D must take active steps to manage their BG levels. This need arises due to the significant health implications of poorly managed BG levels—high BG can induce kidney and eye damage, while low BG may precipitate coma or even death.

Insulin administration, a process that reduces BG levels, is critical for T1D management. However, it is an arduous task requiring injection—either manual or pump-assisted— before each meal and whenever BG levels dangerously escalate. Further complexity stems from the necessity for precise dosage calculation. Over-dosing could trigger hypoglycemia (low BG), while under-dosing could trigger hyperglycemia (high BG), necessitating an accurate balance of insulin. Insulin dosage is calculated based on the amount of carbohydrates in a meal, which is difficult to estimate accurately [Brazeau et al., 2012, Mehta et al., 2009]. Even if meal size is known, other factors such as meal composition make deciding on insulin doses a challenging process, since the body processes carbohydrates differently in the presence of other macronutrients (e.g., fats, proteins) [Bell et al., 2015]. Inaccurate meal size estimates can lead to inappropriate bolus choices, and in turn poor BG management. Additionally, an individual's glucoregulatory system will evolve and develop through adolescence and beyond, meaning that the parameters used to calculate insulin values must be manually updated, presenting an additional burden [The Diabetes Control and Complications Trial Research Group, 1993, Rubin-Falcone et al., 2022].

Technology has catalyzed the development of automated insulin dosing solutions with the goal of making BG management less taxing for patients. Techniques such as Loop [Ahmed et al., 2020], an open-source automatic pancreas system, rely on rule-based BG forecasting models for dosage determination. These methods, however, also require patients to manually select and update parameters, which requires sustained effort as values must be reconsidered whenever an individual's body changes. There are also several proprietary automated BG control algorithms; while the specific forecasting models utilized by these approaches remain undisclosed (they may or may not be rule-based), these approaches depend on patient-managed parameters such as target set point and insulin sensitivity factor [Collyns et al., 2021, Ware et al., 2022].

In contrast to rule-based approaches, researchers have developed machine learning (ML)-based BG forecasting methods that can be automatically updated over time [Marling and Bunescu, 2018a, Fox et al., 2018, Munoz-Organero, 2020]. A BG forecaster is given a brief history of (usually 5-minute interval) BG values (e.g., last hour) and is trained to predict BG 15 minutes to two hours into the future. Insulin, carbohydrates, and other auxiliary signals may also be included as inputs. A machine-learning based forecaster can be retrained at any time on the latest available data, and an accurately trained BG forecaster can be utilized repeatedly to warn a patient prior to an immanent dangerous outcome (hyper- or hypoglycemia), allowing them to take preventive action. Additionally, a BG forecasting model can be utilized to automate insulin dosing by selecting bolus values associated with favorable predictions. While simple rule-based forecasters are often used in control solutions [Ahmed et al., 2020], ML-based forecasters are not reliable enough for wide-spread adoption. Specifically, such models fail to accurately capture the effects of carbohydrates and insulin on BG. We hypothesize that this failure is due to a combination of issues which we address here. In particular, ML-based models assume accurate meal size estimates, do not separate endogenous from exogenous effects, and assume correlations between carbohydrate and insulin values will hold across datasets. Recognizing these limitations, we propose new data-driven approaches to aid in the management of BG.

## 1.1 Challenges, Opportunities & Contributions

Our aim is to discern and address challenges pertaining to the utilization of data-driven approaches for BG management (**Figure 1.1**). We propose (i) solutions to denoising patient-reported meal sizes and (ii) novel approaches to forecasting BG values with sparse but informative variables and correlated input variables. The first contribution is motivated by the fact that current approaches to BG management rely on accurate estimates of meal size (i.e., amount of carbohydrates). Therefore, noisy patient estimates remain a barrier to maximally effective BG management. Our solution provides a more accurate estimate of meal size by utilizing the known relationship between carbohydrate intake and BG. Our novel approach to denoising patient-reported values through leveraging auxiliary signals is described in **Chapter 3**. Our remaining contributions are inspired by a limitation of automated BG management solutions. Specifically, many current control approaches rely on simple rule-based forecasters which require patients to manually manage model parameters. While machine-learning based forecasters could obviate this need via online learning, current deep forecasting approaches struggle to accurately model the effects of insulin and carbohydrates on BG. This is largely due to the relative sparsity of auxiliary variables and correlated inputs. Our solution to the sparsity challenge incorporates domain knowledge to ensure better integration of bolus and carbohydrate values into forecasters. Our "Linked Encoder/Decoder" for improved modeling of sparse informative variables in RNN-based forecasters is described in **Chapter 4**. Our solution to modelling bolus and carbohydrate effects given their correlation leverages the inherent properties of the basal bolus strategy. Our residual-effect

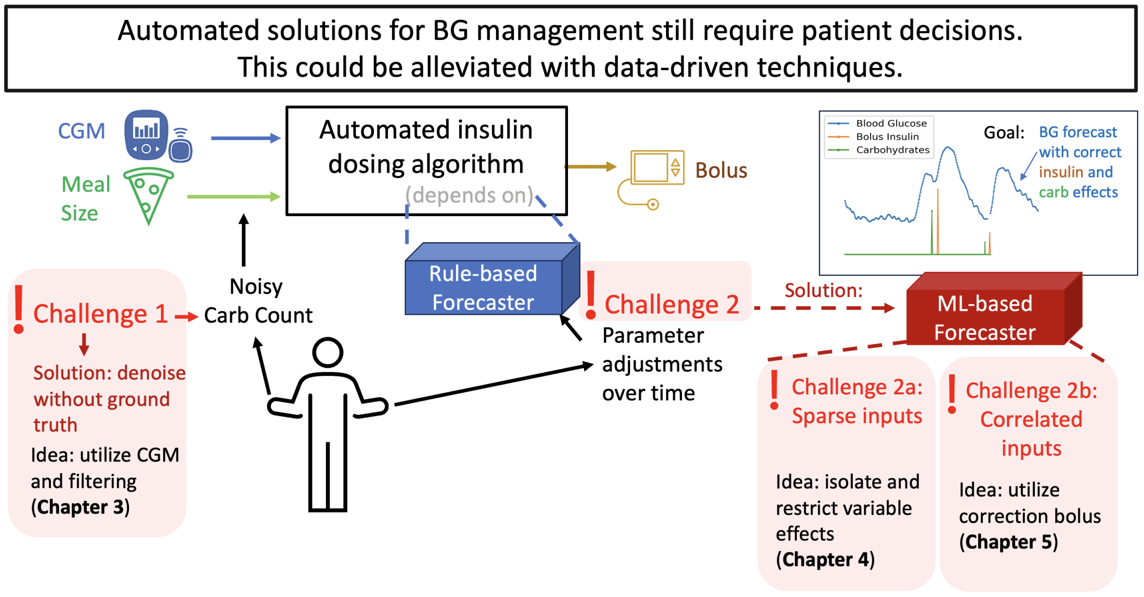

Automated solutions for BG management still require patient decisions.
This could be alleviated with data-driven techniques.
CGM
Meal Size
Automated insulin dosing algorithm
(depends on)
Bolus
Blood Glucose
Bolus Insulin
Carbohydrates
Goal: BG forecast with correct insulin and carb effects
Rule-based Forecaster
Noisy Carb Count
Challenge 1
Solution: denoise without ground truth
Idea: utilize CGM and filtering (Chapter 3)
Challenge 2
Parameter adjustments over time
Solution:
ML-based Forecaster
Challenge 2a: Sparse inputs
Idea: isolate and restrict variable effects (Chapter 4)
Challenge 2b: Correlated inputs
Idea: utilize correction bolus (Chapter 5)

Figure 1.1: In this dissertation, we explore challenges preventing the applicability of data-driven approaches to BG management. Our objective is to learn clean feature representations for noisy measurements and to architect forecasting models capable of accurately deciphering the impacts of carbohydrate intake and insulin boluses on blood glucose levels. To achieve this, we address the challenges posed by the inherently noisy nature of carbohydrate data, the relative sparsity of input variables, the and the strong entanglement between bolus and carbohydrate values.

disentanglement strategy is described in **Chapter 5**. A comprehensive background pertaining to these solutions is delineated in **Chapter 2**. While we focus on BG management as inspiration for this work, each contribution is applicable to multiple domains across healthcare and machine learning more broadly (e.g., denoising patient-reported values; forecasting vital signs, traffic, and stock prices; learning generalizable forecasters for model-based control in robotics settings with correlated movement in adjacent parts).

First, we propose an approach for denoising carbohydrate measurements in the absence of ground truth values. Estimating the carbohydrate content in meals is inherently challenging due to the frequent necessity of this task and its intrinsic complexity [Brazeau et al., 2012, Mehta et al., 2009]. This results in substantial noise and missingness in carbohydrate data, which in turn hampers BG management. Ground truth carbohydrate values are rarely possible to obtain, requiring either a dietitian or pre-packaged meals. Additional factors such as time-of-day metabolism dynamics and meal composition further compound this problem, obscuring the true impact of carbohydrates on BG [Bell et al., 2015]. Existing denoising techniques are unable to address this challenge due to a lack of available ground truth data, low dimensionality, and an unknown noise function. To overcome this challenge, we note that the related and less noisy BG signal can be utilized to recover carbohydrate values. We devise an approach in **Chapter 3** that employs co-teaching [Han et al., 2018] to iteratively filter out the noisiest carbohydrate samples. We then train a denoising autoencoder to refine carbohydrate measurements using the continuous glucose monitoring (CGM) signal and these filtered samples. **This approach is the first to utilize auxiliary input signals and co-teaching to train denoising autoencoders on data where ground truth is unknown**. By incorporating both meal estimates and CGM signals, our approach effectively exploits the relationship between BG and carbohydrates while using patient estimates as an initial hypothesis and as additional data at inference time. Our method does not rely on ground truth carbohydrate values but assumes some meal measurements to be more accurate than others by chance. Our approach could benefit individuals by automatically informing them of when they are over- or under-reporting carbohydrate values, enabling better treatment management in the future.

Next, we identify the *sparse but informative variable* (SIV) problem and propose a novel solution. Bolus and carbohydrate values have a significant impact on BG levels, and accurately modeling their effects would enable a forecaster to be applicable to control applications. Because ML-based forecasters can be autonomously updated, a control system built on such a forecaster would have a clear advantage over current approaches, which require manual parameter management. However, we observe a failure in leveraging meal information in common ML-based forecasting approaches [Rubin-Falcone et al., 2020, Hameed and Klienberg, 2020, McShinsky and Marsha, 2020], making them inapplicable to control. We suspect this is due to the sparse presence of these variables relative to the target variable. Non-zero bolus and carbohydrate values occur about 3-4 times daily, while BG is logged every

5 minutes. This situation differs from the sparsely sampled variable problem: while bolus and carbohydrates are measured as frequently as BG, their true values are predominantly zero. We make use of domain knowldege to overcome this sparsity problem. In **Chapter 4**, we tackle this SIV problem with a novel recurrent neural network (RNN) encoder/decoder strategy, isolating SIV dynamics with an auxiliary decoder and constraining bolus and carb effects to align with domain knowledge. **Our method is the first to explicitly address the SIV problem. Our approach uses a minimal amount of supervision to incorporate domain knowledge, unlike other forecasting approaches that either require highly restrictive models or exclude domain knowledge altogether**. This approach not only enhances overall forecast accuracy but also augments the impact of bolus and carbohydrate variables on the model.

Finally, we strive to ensure out-of-sample generalizability across BG control strategies, even when bolus and carbohydrate values are highly confounded in the training data. The basal bolus scheme is the most commonly recommended blood glucose control strategy [Janez et al., 2020]. With this scheme, a consistent amount of insulin is administered throughout the day (basal), and larger quantities are required during meals (bolus). Bolus values are calculated as a linear combination of the carbohydrate content in a meal and current BG levels, resulting in entangled input features. This entanglement may lead forecast models to learn representations that perform well on basal bolus data (the behavior policy) but fail to accurately capture the true effects of the bolus and carbohydrate values, thus hampering out-of-sample generalizability. Therefore, ML-based forecasters trained on basal bolus data cannot be easily applied to control approaches. In **Chapter 5**, we leverage the known properties of the basal bolus strategy to isolate the bolus's independent effect on BG and train the model accordingly. In doing so we more accurately model the independent effects of carbohydrates and bolus insulin on future blood glucose values. This in turn yields better predictions during planning in model-based control, potentially allowing automatically adjustable ML-based forecasters to be applied directly to BG management. Our approach is inspired by techniques for estimating average treatment effects in the presence of confounding [Rubin, 1974, Rosenbaum and Rubin, 1983], although a lack of overlap makes these techniques non-applicable. Our approach bears similarity to S-learner [Künzel et al., 2019], where treatment effect is estimated by removing a related variable from the model, and experimental treatment effect transfer techniques, where a small amount of non-confounded data are used to guide training on confounded data [Kallus et al., 2018, Hatt et al., 2022]. **Our approach utilizes the variations within the training data to isolate a partial bolus effect, and then trains the model to learn a proportional effect**. This facilitates learning a model that can generalize out-of-distribution (to control strategies without confounded bolus values) without requiring additional training. We aim to create a model that can be trained on a small amount of entangled data and then directly applied to a model-based control application, where evaluation on unentanlged data would be necessary.

## 1.2 Summary of Contributions

In summary, we make the following contributions:

- In Chapter 3, we propose a technique for reconstructing noisy data in the absence of known ground-truth samples by leveraging auxiliary data streams and iteratively filtering the dataset. Our approach successfully recovers accurate carbohydrate values, potentially leading to enhanced BG control.
- In Chapter 4, we introduce the *sparse but informative variable* (SIV) problem. This problem arises when auxiliary input variables, though infrequent in occurrence, have substantial influence on the target variable to be forecasted, and RNN-based approaches struggle to incorporate them effectively. We propose a solution that mitigates this limitation, and we demonstrate that it enhances BG forecasting accuracy by better utilizing carbohydrate and bolus values, both of which are SIVs.
- In Chapter 5, we highlight that the most common BG management strategy calculates

bolus values as a linear combination of BG and carbohydrate values. This induces a strong correlation between carbohydrates and bolus inulin in training data and impedes a forecasting model's ability to learn the independent effects of boluses and carbohydrates on BG values. We propose a solution that utilizes the known properties of the control strategy to learn a generalizable model from entangled inputs.

**Generalizable Insights for Machine Learning.** While each chapter of this dissertation addresses a unique barrier to simplified BG management, each contribution has implications for other machine learning applications. Our denoising technique can be applied to other situations involving patient-reported outcomes paired with sensor data, such as mood scores paired with wearable-recorded step and sleep data. SIVs are present in various medical settings, such as blood pressure forecasting (where vasopressor/vasodilator administrations serve as SIVs), as well as in non-medical domains like travel forecasting (with holidays and major events as SIVs), and stock price prediction (where quarterly reports and news releases are SIVs). Finally, our approach to disentangling exogenous effects in forecasting could be beneficial wherever two input variables are confounded to various degrees, such as in electric load forecasting, where two power sources may exhibit strong correlation. Our approach could also be useful in any model-based control setting where system variables are correlated, for example, in robotics, where the movement of adjacent parts will be correlated but isolating their individual impacts on the system could be crucial. **In sum, by developing methods for assisting in BG management in T1D, we advance towards improved patient outcomes in this health-critical application and generate valuable insights into challenges prevalent in various other machine learning applications.**

The rest of the thesis is organized as follows. In Chapter 2, we describe relevant background concepts used throughout the remainder of the thesis. Chapters 3, 4 and 5 summarize the details of our contributions. The concluding chapter (Chapter 6) reflects on future directions in relation to the work presented in this dissertation.

CHAPTER 2

# Background: Blood Glucose Management in Type 1 Diabetes

In this chapter, we cover topics referenced throughout this dissertation in the fields of blood glucose (BG) management in type 1 diabetes (T1D), time-series forecasting, and model-based control.

## 2.1 Manual Blood Glucose Management

Individuals with T1D experience an impairment in their glucoregulatory system, which renders their bodies incapable of automatically maintaining healthy BG levels, a process that occurs naturally in healthy individuals. T1D affects millions of people around the globe [Gregory et al., 2022]. The management of this condition necessitates manual control of BG levels through regular insulin administration. This process poses a significant burden as it necessitates multiple daily insulin injections and requires precise estimation of meal sizes for accurate dosage calculation.

### 2.1.1 The Basal Bolus Strategy

The most widespread BG control strategy is the basal bolus method [Janez et al., 2020], where a continuous dose of slow-acting (basal) insulin is delivered to maintain consistent BG levels, supplemented by post-meal bolus injections to correct any potential BG surges.

Bolus values $b$ are calculated based on current BG levels $g$ and meal size estimates $c$ as: $b = \frac{c}{CF} + \mathbf{1}_{\{g>TG\}}\frac{g-TG}{CR}$, where patient-specific parameters carbohydrate ratio ($CR$), correction factor ($CF$), and target glucose ($TG$) are estimated to match insulin sensitivity to meal size, insulin sensitivity to BG levels, and an upper bound to the healthy BG range, respectively. Other manual control strategies share a similar core principle, wherein insulin doses are calculated based on current BG levels and estimated meal size [Weinstock, 2023].

### 2.1.2 Measuring Success in BG Management

The effectiveness of a BG management strategy can be evaluated using various metrics including risk scores that gauge the safety of current BG levels [Magni et al., 2007]. Although these metrics can be derived from a single BG reading, a more comprehensive assessment such as time in range (TIR), becomes feasible with additional data. TIR represents the percentage of time an individual's BG levels remain within the healthy, or euglycemic, range and can be accurately measured with extended data sets, such as a week's worth of 5-minute BG measurements.

### 2.1.3 Meal Estimates

One of the major hurdles in BG management is estimating the carbohydrate content of each meal. The estimation process is challenging and often leads to an inaccurate meal signal and inappropriate boluses [Brazeau et al., 2012, Mehta et al., 2009]. In addition to misestimation, there are other factors contributing to the disparity between recorded carbohydrate intake and its actual effect on BG. For instance, the potential variation in meal types is often inadequately captured. This is problematic as the impact of carbohydrates on BG can be moderated by factors such as the speed at which a meal is consumed or the proportion of protein, fat, and other nutrients present [Bell et al., 2015]. Additionally, the precise timing of a meal might not be accurately recorded. These considerations add further layers of complexity to the use of carbohydrate data as a signal in forecasting or control approaches, even when the carbohydrate count is meticulously documented. In an unsupervised learning context, denoising techniques could help derive more accurate representations of carbohydrate data, taking into account these additional sources of variability. These refined representations could be more pertinent for BG management than the exact amount of carbohydrates consumed, thereby potentially enhancing the performance of forecasting and control algorithms. Additionally, these denoising techniques have potential utility towards patient education, in that automated utilization of these approaches could aid individuals in understanding and correcting systematic errors in estimated carbohydrate counts.

## 2.2 Automated Blood Glucose Management

The advent of modern continuous glucose monitors (CGMs), insulin pumps, and smartphone technology has facilitated the development of automated control strategies by making real-time communication between devices possible. The DIY community has been employing model-based approaches for over a decade [Ahmed et al., 2020], exemplified by LOOP, a system that uses a basic linear model to forecast glucose trajectories based on bolus and carbohydrate effects. Commercially available automated solutions also exist [Collyns et al., 2021, Ware et al., 2022]. Each of these approaches require meal announcements and rely on an underlying model or forecaster of BG dynamics, which is used to select the best performing bolus value. While these approaches require manual parameter updates, machine-learning-based solutions could obviate this need through automatic adjustments. It is feasible to control BG levels using model-free approaches that operate without a forecaster, but these methods demand extensive training data (e.g., years of CGM measurements), which renders them impractical [Fox et al., 2020]. Model-based control strategies offer an alternative since they are oftentimes more sample efficient [Atui, 2015]. Machine learning-driven model-based control solutions could significantly streamline BG management.

### 2.2.1 Model-Based Control

In control-based settings, an agent learns to interact with an environment by taking some action with the goal of optimizing some reward. In the simplest form of model-based control [Argenson and Dulac-Arnold, 2021], a model of the environment (i.e., a forecaster) is paired with a planning algorithm. For example, in random shooting, a model or forecaster is utilized to create predictions for multiple potential actions and the action with the highest predicted reward is selected. With this approach a forecaster may be trained prior to control using whatever data are available, but it will not be guaranteed to perform well outside of the training data distribution.

#### 2.2.1.1 Forecasting

Blood glucose forecasting plays a pivotal role in the management of type 1 diabetes (T1D) and has been extensively investigated in a variety of settings including deep learning [Rubin-Falcone et al., 2020, Fox et al., 2018, Munoz-Organero, 2020]. The goal of BG forecasting is to estimate future BG concentration within a defined predictive horizon based on a historical record of BG measurements and other pertinent input signals. This is a challenging task because glucose dynamics vary based on factors including activity, time of day, and hormone levels, resulting in significant non-stationarity. Additionally, many factors create significant heterogeneity in BG dynamics across individuals [Redondo and Morgan, 2023], making the training of patient-specific forecasters preferable. Accurate BG forecasting models aim to predict potential BG highs or lows, thus providing individuals with advanced warning and ample time to take preventative action. The field of BG forecasting has seen a broad spectrum of methodologies, from classical approaches like ARIMA [Yang et al., 2019] and support vector regression [van Doorn et al., 2021], to modern deep learning techniques encompassing RNNs [Mirshekarian et al., 2019, Rabby et al., 2021], attention models [Mirshekarian et al., 2019, Armandpour et al., 2021], simple MLPs [van Doorn et al., 2021], and residual networks [Rubin-Falcone et al., 2020]. Physiological models of the blood glucose system in T1D have been in development for many years [Palumbo et al., 2013], and some approaches utilize these models for forecasting [Hameed and Kleinberg, 2022]. Recent approaches have also combined deep approaches with these physiological models [Miller et al., 2020]. Although many BG forecasting models are univariate [Rodriguez, 2021], there has been research exploring the incorporation of auxiliary variables such as insulin and carbohydrate intake, time of day, and heart rate information. Despite these auxiliary variables providing some degree of forecast performance enhancement, the effect is often marginal [Rubin-Falcone et al., 2020, Hameed and Klienberg, 2020, McShinsky and Marsha, 2020]. While this discrepancy has not been thoroughly explored, forecasting with more than one input variable is a well studied problem more generally [Rockwell and Davis, 2016, Chatfield, 2000] and including multiple variables when aiming for univariate prediction is often beneficial. Many techniques have been proposed to learn inter-variable relationships in forecasting tasks, including attention-based approaches [Qin et al., 2017, Pantiskas et al., 2020], normalizing flows in probablistic settings [Rasul et al., 2020, Emmanuel de Bezenca an et al., 2020] and explicit modeling of inter-variable relationships [Gu et al., 2020, Freiburghaus et al., 2020, Pantiskas et al., 2020, Cao et al., 2020, Xu et al., 2020a]. These approaches do not utilize domain knowledge, but there has been other work in forecasting that combines deep learning with domain knowledge to reduce the hypothesis space [Munoz-Organero, 2020, Miller et al., 2020, Huanga et al., 2014].

Generally, forecasters are trained and evaluated using data from the same distribution. They are often assessed on training data that has been held out for testing purposes, where lower error—measured by metrics such as mean square error or mean absolute error—is preferred. However, while these metrics effectively gauge a forecaster's performance in a static environment, they are not always reliable for evaluating its suitability for model-based control. In many model-based control scenarios, a forecaster is required to provide accurate predictions across a broad spectrum of potential action values. This requirement leads to a mismatch between the distributions of training and evaluation data. Consequently, new evaluation methodologies are essential to thoroughly evaluate forecasters, especially when considering their adaptability to control scenarios.

#### 2.2.1.2 Planning

Planning can be described in the context of a Markov decision process where an agent selects actions based on an observed state. After an action is chosen, a new state occurs, determined by a transition function that captures environmental dynamics. The effectiveness of an agent's actions is measured by a reward assigned at each timepoint. In model-based planning, the action selection is guided by a model (a forecaster) of the transition function, aiming to achieve the best predicted reward. For BG management, this reward could be a measure of glycemic control or TIR. In the simplest model-based planning scenario, the model is trained on existing data. A predefined planning strategy, such as random shooting, then uses this model to select the action that leads to the best outcome or highest reward. Notably, the planning algorithm itself does not need to be learned; it can be predetermined, meaning that no exploration is necessary when applying the algorithm for control purposes.

# CHAPTER 3

# Denoising Autoencoders for Learning from Noisy Patient-Reported Data

## 3.1 Introduction

With the increasing ubiquity of wearable sensor technology (e.g., fitbit), there has been an explosion in the number of studies seeking to correlate data from these technologies with patient-reported data (e.g., near-falls), with the goal of remote patient monitoring [Kious et al., 2019, Hauth et al., 2021]. Patient-reported outcomes have been used in studies of cancer treatment [Nguyen et al., 2020], multiple sclerosis [D'Amico et al., 2019], diabetes [Wee et al., 2021], and mental health [McIntyre et al., 2022], and are also used to quantify patient-experience of care [Bull et al., 2019]. However, patient-reported data are often noisy and difficult to validate [Churruca et al., 2021]. The accuracy of these data may change day-to-day or even hour-to-hour [McKenna, 2011], making it challenging to detect meaningful changes over time [van der Willik et al., 2020]. Moreover, in many cases, ground truth is difficult if not impossible to obtain. Current state of the art denoising approaches utilize assumptions that do not hold in this setting (e.g., high-dimensional data, known noise function). In light of these limitations, we aim to develop an approach that can denoise patient-reported data and increase their utility in downstream tasks (e.g., adverse outcome prediction). Our approach is based on the observation that while ground truth values for the target variable may be unavailable, other more reliable data streams (i.e., data collected

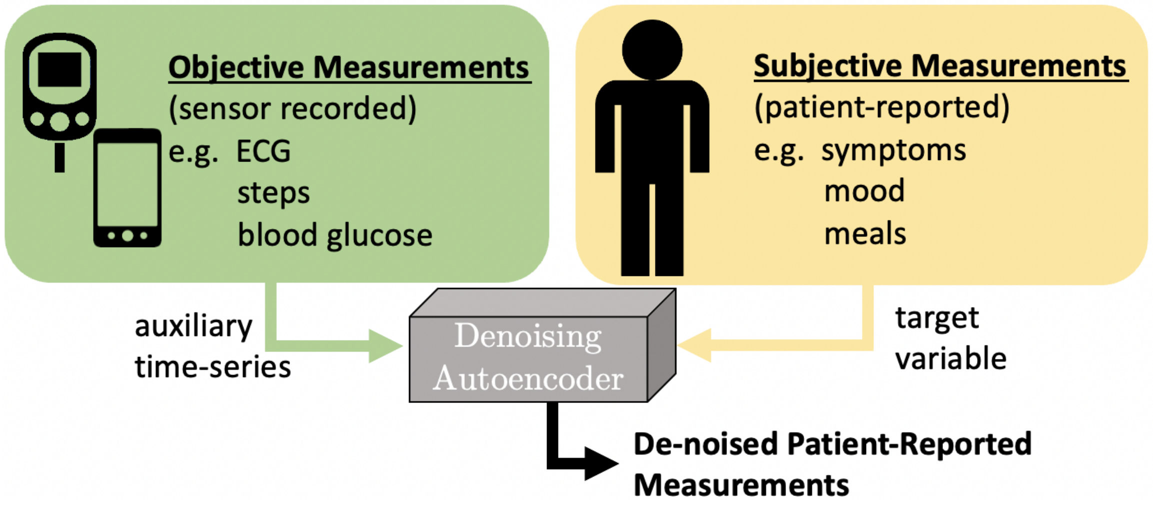


Figure 3.1: **Overview**. In our setting, given access to both subjective patient-reported data and higher-fidelity data from wearable sensors, we aim to denoise subjective measurements.

from wearable sensors) are often collected alongside noisy patient-reported measurements. We hypothesize that these more reliable related data streams can help in recovering the noisier variables (**Figure 3.1**).

In this chapter, as throughout this dissertation, we take inspiration from BG management in T1D. BG, when measured by a continuous glucose monitor (CGM), has relatively little noise [Shah et al., 2018], while the amount of carbohydrates in a meal are patient-reported and subject to error [Brazeau et al., 2012, Mehta et al., 2009]. Recognizing this variation in the level of noise across signals, we propose an approach that utilizes objective measurements (e.g., BG) to update noisy patient-reported data. This retrospective correction could help in downstream tasks: patients can learn when they are over- or under-reporting and adjust in the future, ultimately improving disease management. Training forecasters with denoised carbohydrate values could enable them to better learn the effects of both carbohydrates and insulin (by reducing noise-influenced confounding), leading to better applicability to model-based control approaches.

Denoising autoencoders (DAEs) [Vincent et al., 2008] have been used to accurately denoise signals, including medical images [Gondara, 2016], ECG signals [Xiong et al., 2016], and power system measurements [Lin et al., 2019]. However, this approach generally requires access to both patient-reported measurements and ground truth measurements at training. In many real-world settings (including ours), only patient-reported target samples are available at training (we don't have access to paired ground truth patient-reported data). Work in computer vision has addressed this problem through extensions that either require paired noisy samples for each data point (e.g., multiple images of the same object) [Lehtinen et al., 2018] or rely on patch-based analysis [Krull et al., 2018, Laine et al., 2019, Xie et al., 2020, Batson and Royer, 2019]. Similar approaches do not extend to patient-reported data, where paired samples rarely exist and patch-based techniques do not apply due to a lack of spatial feature dependencies. Others have proposed techniques that leverage knowledge of the noise distribution to recover the clean signal, but their applicability is limited in our setting, as noise for patient-reported variables is rarely weak or known [Kim and Ye, 2021, Moran et al., 2019, Xu et al., 2020b]. In contrast to prior work that has focused on missingness in patient-reported datastreams (including meal reports in diabetes management), we focus only on denoising existing measurements.

We adapt denoising autoencoders (DAEs) for patient-reported data. Our approach, 'Noise$^{+}$2Noise', learns to denoise a target signal (e.g., patient-reported meals) given only potentially noisy target samples (without access to ground truth) and an auxiliary clean signal (e.g., BG measurements). Inspired by work in image denoising [Lehtinen et al., 2018, Xu et al., 2020b], our approach augments existing DAEs with an auxiliary signal, leveraging the relationship between the auxiliary and target signals. In addition, we adapt a novel co-teaching approach from the noisy label literature [Han et al., 2018] to train two DAEs. Our approach works by iteratively selecting lower-noise target samples for training. Through a case study in blood glucose management, we demonstrate that our proposed approach can more accurately recover patient-reported data in the presence of noise compared to several baselines. Our contributions are as follows:

•We formalize an important problem in remote patient monitoring related to denoising patient-reported data.

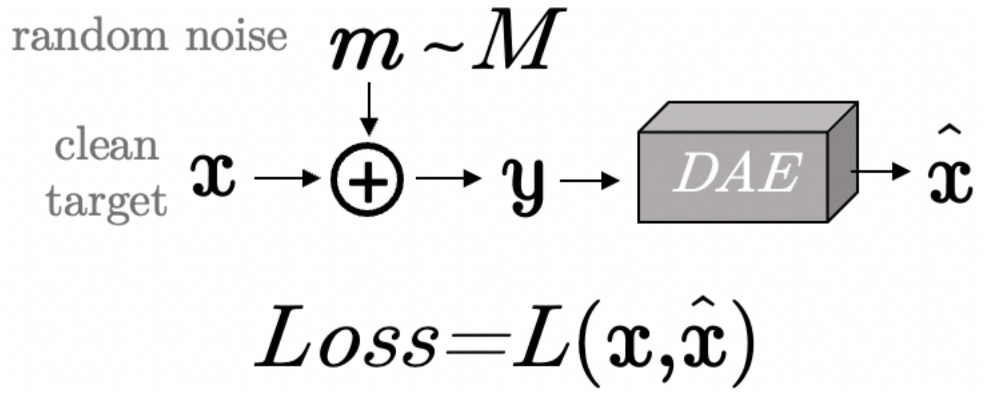


Figure 3.2: In vanilla DAE training, access to a clean target identified a priori to training ($\mathbf{x}$) is assumed. Noise ($\mathbf{m}$) is added to $\mathbf{x}$ to create noisy sample $\mathbf{y}$, and the DAE is tasked with reconstructing $\mathbf{x}$.

•We propose a novel approach based on DAEs that leverages an auxiliary low-noise signal to denoise a target variable without access to ground truth target data.

•We demonstrate improved denoising of carbohydrate values for blood glucose management compared to baselines in a simulated data setting.

•We propose and validate a proxy measure for evaluating carbohydrate denoising when ground truth is unavailable. Our approach outperforms baselines on a real-life dataset based on this metric.

## 3.2 Background and Related Work

Our approach takes inspiration from work in DAEs, commonly used in image denoising, and work in in noisy label learning. Below we briefly provide background on these topics.

In DAE training, a model input is corrupted and a network is tasked with recovering the original input (**Figure 3.2**). In this way, the network cannot learn the identity, unlike in basic autoencoder training [Vincent et al., 2008]. Recent work has focused on using DAEs to recover clean signals from only noisy signals. The vast majority of this work lies in image analysis and builds off of “noise2noise” [Lehtinen et al., 2018], an approach that uses multiple noisy instances of the same image to learn to denoise the image. The approach relies on the fact that if the noise is zero mean, using a secondary noisy instance (besides the input image) as a target will produce a network that learns the clean image, in expectation, when enough training data are available. When paired samples are unavailable, other approaches exploit patches sampled from the image [Laine et al., 2019, Xie et al., 2020, Batson and Royer, 2019], but these do not apply to our setting since the target data we aim to correct are univariate. Other approaches eschew relying on inter-variable relationships but rely heavily on a known noise function [Moran et al., 2019, Kim and Ye, 2021] or a low expectation and variance noise function [Xu et al., 2020b]. Our approach builds off Xu et al. [2020b], learning to reconstruct a signal from only potentially noisy samples of that signal, but in contrast to Moran et al. [2019] or Kim and Ye [2021], we do not make strong assumptions about the noise distribution. Instead, we leverage an auxiliary signal and iteratively filter out noisy samples.

Our approach is, in part, related to work in noisy-label learning, where a common approach involves identifying and reweighting samples with clean labels during training. Samples are filtered based on gradient values [Ren et al., 2020], Jacobian ranking [Mirzasoleiman et al., 2020] or some latent state [Lee et al., 2019, Wu et al., 2020]. Co-teaching [Han et al., 2018], which builds off of mentor net [Jiang et al., 2018], filters out incorrectly labeled samples by utilizing two networks in parallel. For each network, backpropagation is performed using only samples within the current mini-batch for which the loss of the *other* network is lowest. Intuitively, samples with incorrect labels are likely to have higher loss and therefore be removed. Using two networks in parallel provides robustness to outliers and initially misclassified samples, to which single-network boosting-style approaches are sensitive. To date, these approaches have been primarily explored in supervised and semi-supervised settings. In contrast, we consider an unsupervised setting in which ground truth labels are unavailable and, instead, the input signals themselves are corrupted. To the best of our knowledge, such a co-teaching approach has not been explored in the context of denoising.

## 3.3 Problem Setup

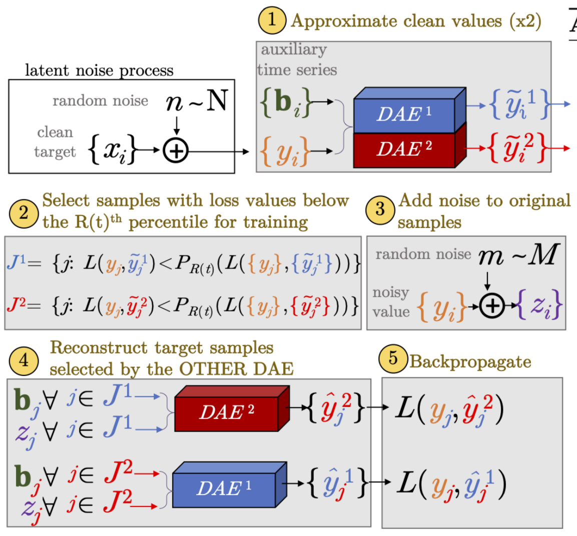
Given dataset $\mathcal{D}$ with $k$ samples $\mathcal{D} = \{y_i, \mathbf{b}_i\}_{i=1}^{k}$, where $y_i \in \mathcal{R}$ denotes a noisy target sample and $\mathbf{b}_i \in \mathcal{R}^T$ denotes an auxiliary time-series. True target values, $\{x_i\}_{i=1}^{k}$, are unknown. For each sample, $y_i = x_i + n_i$, where $n_i$ denotes a random variable drawn from some unknown distribution, i.e., $n_i \sim \mathcal{N}$. We aim to learn some mapping: $f : (y, \mathbf{b}) \rightarrow x$, given $\mathcal{D}$.

We assume that the distribution $\mathcal{N}$ is independent of both $\mathbf{b}$ and $y$. We assume that $\mathbf{b}$ is related to $x$, such that some approximate mapping $\mathbf{b} \rightarrow x$ exists. Implicitly, we assume that the timing of the target variable, relative to $\mathbf{b}$, is fixed and that the ordering is such that a mapping is possible (i.e., if $x$ causes a change in $\mathbf{b}$, then $\mathbf{b}$ values must follow $y$ in time so $x$ can be learned retrospectively). We also assume that some values of $n$ are near zero such that within training data $\mathcal{D}$, there exists a subset $S$ with sufficient size for training such that the mean and variance of $n_s \forall s \in S$ are negligible compared to the mean and variance of $x_s \forall s \in S$. We assume that the distribution of low noise samples is such that they cover all regions of the input; *i.e.* that $S$ does not exclude entire regions in the range of possible $y$ values. Finally, we assume that the relationship between $\mathbf{b}$ and $x$ can be accurately captured with a recurrent neural network (RNN).

While we note that these assumptions allow for noise of arbitrary average magnitude, access to some low-noise (though unlabeled) samples is assumed. Similar assumptions are common in healthcare applications [Chang et al., 2020, Geng et al., 2022, Zhang et al., 2021], where access to a small number of low noise or ground truth samples (through data curation) is possible but labor intensive and costly (e.g., prospective data collection or clinician review).

## 3.4 Methods

First, we will describe our proposed training regime for addressing noisy patient-reported values. Then we will briefly describe our evaluation set up.



1 Approximate clean values (x2)
latent noise process
random noise $n \sim N$
clean target $\{x_i\}$
auxiliary time series
$\{\mathbf{b}_i\}$
$\{y_i\}$
$DAE^1$
$DAE^2$
$\{\tilde{y}_i^1\}$
$\{\tilde{y}_i^2\}$
2 Select samples with loss values below the R(t)th percentile for training
$J^1 = \{j\text{: } L(y_j,\tilde{y}_j^1) < P_{R(t)}(L(\{y_j\},\{\tilde{y}_j^1\}))\}$
$J^2 = \{j\text{: } L(y_j,\tilde{y}_j^2) < P_{R(t)}(L(\{y_j\},\{\tilde{y}_j^2\}))\}$
3 Add noise to original samples
random noise $m \sim M$
noisy value $\{y_i\}$
$\{z_i\}$
4 Reconstruct target samples selected by the OTHER DAE
$\mathbf{b}_j \forall j \in J^1$
$z_j \forall j \in J^1$
$DAE^2$
$\{\hat{y}_j^2\}$
$\mathbf{b}_j \forall j \in J^2$
$z_j \forall j \in J^2$
$DAE^1$
$\{\hat{y}_j^1\}$
5 Backpropagate
$L(y_j,\hat{y}_j^2)$
$L(y_j,\hat{y}_j^1)$

```
Algorithm   Noise$^+$2Noise: Co-taught Denoising Auto-Encoders
Input DAE$^1$ and DAE$^2$ with randomly initialized weights,
    Loss function $L$, noise distribution $M$, fixed $\tau$, epoch $E_k$,
    epoch $E_{max}$
for $t = 1, 2, \ldots, E_{max}$ do
    Fetch batch $\{y_i\}$ and $\{\mathbf{b}_i\}$
  1 $\{\tilde{y}_i^1\} = DAE^1(\{\mathbf{b}_i\}, \{y_i\})$          \\Approximate
    $\{\tilde{y}_i^2\} = DAE^2(\{\mathbf{b}_i\}, \{y_i\})$           clean values
  2 $J^1 = \{j : L(y_j, \tilde{y}_j^1) < P_{R(t)}(L(\{y_j\}, \{\tilde{y}_j^1\}))\}$    \\Select
    $J^2 = \{j : L(y_j, \tilde{y}_j^2) < P_{R(t)}(L(\{y_j\}, \{\tilde{y}_j^2\}))\}$     samples
  3 Sample $\{m_i\} \sim M$
    $\{z_i\} = \{y_i + m_i\}$                      \\Doubly noisy values
  4 $\{\hat{y}_j^1\} = DAE^1(\{\mathbf{b}_j\}, \{z_j\})$ for $j \in J^2$     \\Reconstruct
    $\{\hat{y}_j^2\} = DAE^2(\{\mathbf{b}_j\}, \{z_j\})$ for $j \in J^1$       noisy signal
  5 BACKPROP $L(y_j, \hat{y}_j^1)$ for $j \in J^2$ and $L(y_j, \hat{y}_j^2)$ for $j \in J^1$
    Update $R(t) = 1 - min\{\frac{t}{E_k}\tau, \tau\}$
Output $DAE^1$ and $DAE^2$
```

Figure 3.3: 'Noise$^+$2Noise'. Sample selection is performed with each DAE's output when given the uncorrupted $y$ signal, but backpropagation is performed on the model's output when given a corrupted $y$ signal. The loss values of $DAE^1$ are used to select the sample for backpropagation for $DAE^2$ and vice versa. $P_{R(t)}$ denotes the $R(t)^{th}$ percentile, where $R(t)$ is a function of iteration $t$. $\mathbf{b}$ signals are aligned so that non-zero $y$ values always occur exclusively at the first position in the input window. The value of y is passed through to the decoder at each timepoint in a separate channel from $\mathbf{b}$.

### 3.4.1 Proposed Approach

**Overview.** Our method, 'Noise$^+$2Noise' ($\mathbf{N^+2N}$), is summarized in **Figure 3.3**. At a high level, we filter out noisy samples during training, refining the model parameters on selected samples of $y$ (noisy carbohydrate values) estimated to have the least noise. To identify these higher fidelity samples within a batch, we identify the subset of samples with loss values below the $R(t)^{th}$ percentile of the batch, where $R(t)$ is an increasing function of iteration $t$. These 'low-noise' samples are augmented with additional noise and, along with corresponding $\mathbf{b}$ (BG) vectors, are input to the DAE, which outputs a reconstruction $\hat{y}$. We then backpropagate using the squared error between $\hat{y}$ and $y$. This corresponds to training on the samples estimated to have the least noise, while utilizing the noisy signal as input. To increase robustness, we consider an ensemble approach in which we use co-teaching to train two DAEs ($DAE^1$ and $DAE^2$). During training, the samples identified as low-loss by $DAE^1$ are augmented and passed to $DAE^2$ for backpropagation, and vice-versa.

**Denoising Autoencoder.** In our setup, additional noise ($m$) is added to $y$ to produce $z$, a 'doubly' noisy measurement of $x$ (hidden ground truth carbohydrate values). $z$ and $\mathbf{b}$ are input to a network (henceforth denoted DAE) that outputs $\hat{y} = DAE(z, \mathbf{b})$, and the network is trained to reconstruct $y$: loss is measured between $y$ and $\hat{y}$. As shown by Xu et al. [2020b], when the expectation and variance of the noise distribution $N$ are negligible compared to those of the signal, the model parameters that minimize the loss between $\hat{y}$ and $y$ are very close to the optimal parameters of a model trained on data that is identified as clean prior to training.

**Co-teaching DAEs.** We do not expect the noise value to be below a certain threshold at all times, but we do assume that some of the samples will have noise close to zero. We identify and train using these samples via an adapted co-teaching approach [Han et al., 2018]. We utilize two DAEs, and for each, we backpropagate using only the samples for which the denoised $y$ values from the other DAE are near the original $y$ values. If the denoised $y$ values approach $x$, we are then selecting samples for which the estimated noise $n$ is lowest.

*Claim*: When using co-teaching to train two DAEs ($DAE^1$ and $DAE^2$) in parallel, denoised $y$ values $\tilde{y}^1 = DAE^1(y, \mathbf{b})$ and $\tilde{y}^2 = DAE^2(y, \mathbf{b})$ approach $x$. *Justification*: Based on the main result of Xu et al. [2020b], if the two DAEs are trained in a standard fashion, $\tilde{y}^1$ and $\tilde{y}^2$ converge to approximately $x$ if, in the training data, the expectation and variance of the signal are much greater than those of the noise. We have assumed that such a sub-sample exists in our dataset, and propose that co-teaching is likely to select such a sub-sample. Because the noise $n$ is independent of both $y$ and $\mathbf{b}$, only the true component of the signal can be learned during training, so the models will learn some function of $x$. The intuition motivating Han et al. [2018] is that, in a noisy-label-learning setting, small-loss instances occur either when both the model and label are correct, OR when the model has memorized an incorrect label. By gradually decreasing the sample size of the training data based on loss values at each iteration, noisy label samples are dropped before the model can memorize them. In a similar vein, in our setting, a model will output a cleaned value $\tilde{y}$ that is close to $y$ when either it has learned the correct function of $x$ and $y$ is near to $x$, or if the model has memorized part of the noise. By employing the co-teaching sample selection method, we believe that the model selects the clean samples before it can memorize the noisy ones. We note that it is not impossible for the model to learn a biased function of $x$, but in practice, we have found that this approach works well even when there is fairly substantial bias in the noise.

**Sample Selection.** We select the samples with the lowest estimated noise for backpropagation. If $\tilde{y}^1$ and $\tilde{y}^2$ approach each network's estimated value of $x$, then $DAE^1$'s estimate of $n$, the noise between $x$ and $y$, is approximately $y - \tilde{y}^1$ (and similar for $DAE^2$). For loss function $L$, we use $L(\tilde{y}^1, y)$ and $L(\tilde{y}^2, y)$ to select samples. As in Han et al. [2018], we begin by training on the full sample. Over the course of training, as the DAEs are expected to become more accurate, we gradually reduce the sample. Hyperparameter $\tau \in (0, 1)$ in the pseudocode of **Figure 3.3** represents the maximum proportion of samples removed and $E_k$ represents the iteration at which we stop increasing the proportion of samples removed.

A linear decrease in sample size as a function of iteration $t$ is implemented by using the lowest-loss $R(t) = (1 - Maximum(\frac{t}{E_K}\tau, \tau)) \cdot 100\%$ of samples for backpropagation.

**Training.** Each DAE is trained on the samples for which the *other* network estimates that the noise is lowest, which prevents sensitivity to error propagation from wrongly selected samples early in training. Samples selected by $DAE^1$ ($y_j$ where $j \in J^1$) are augmented with additional noise $m_j \sim M$ to generate $z_j$ values. $z_j$, along with corresponding $\mathbf{b}_j$ vectors, are input to $DAE^2$, which outputs a reconstruction of $y_j$: $\hat{y}_j^2$. We then backpropagate using the squared error between $\hat{y}_j^2$ and $y_j$. Similarly, we only use samples $y_j$ where $j \in J^2$, augmented with $m_j \sim M$, to backpropagate $DAE^1$. By selecting samples based on $L(\tilde{y}, y)$ rather than based on $L(\hat{y}, y)$, we are able to select a sample independent of secondary noise value $m$. Selecting samples dependent on $m$ would be confounding because samples might then be selected based on how low the value of $m$ is at the current iteration, rather than the value of $n$, which is hidden. Back-propagation is performed on an input that does not include unaltered $y$ values, so the model is not likely to learn the identity function. Sample selection is always performed by the other DAE, so compared to boosting or other one-network approaches, our method is less sensitive to error propagation from wrongly selected samples early in training.

**Co-teaching+.** We utilize co-teaching+ [Yu et al., 2019], where samples for which the models disagree are selected for backpropagation. As a result, each model learns from the samples for which the other model's estimates were better. This prevents the models from learning from the samples that they agree upon, which prevents convergence [Yu et al., 2019], maintaining unique strengths in each model. We remove the $\sigma\%$ of samples for which the models' outputs are closest ($\sigma$ is a hyperparameter). This step is performed prior to the sample selection step: the $\sigma\%$ of samples for which the distance between $\hat{y}^1$ and $\hat{y}^2$ are lowest are removed, and then the remaining samples for which $L(\tilde{y}^1, y)$ is lowest are used for the backpropagation of $DAE^2$ and vice versa.

### 3.4.2 Experimental setup

We evaluate our approach in the context of learning to correct noisy patient-reported carbohydrate measurements. We compare performance to several baselines across real and simulated datasets.

#### 3.4.2.1 Datasets

We utilize two type 1 diabetes-based datasets. The simulated dataset provides access to ground truth to which we can directly compare our method's denoised outputs. The real dataset provides a more challenging setting for quantifying the efficacy of our approach, but corresponds to real-world scenarios. Both datasets are publicly available and have been previously explored in the context of forecasting and control [Man et al., 2014, Xie, 2018, Marling and Bunescu, 2018b, 2020b]. Both datasets consist of blood glucose, bolus (fast-acting) insulin, basal (slow-acting) insulin, and carbohydrate values. All variables were scaled to be between zero and one. For both datasets, time-series trajectories for each patient were split into windows of 2 hour length ($T = 24$ 5-minute time points). We ignore windows where a carbohydrate occurs in anywhere but the first position, using only windows with no carbohydrates or carbohydrates at the beginning of the window during training. This means we also ignore windows with more than one carbohydrate present. In a real-world setting these values could be updated recursively, but we simplify our setting here. The auxiliary signal is assumed to be cleaner than the highly noisy target variable, but not completely noise-free. In practice, the auxiliary signal can have noise- there is approximately 5% noise in both the simulated and real blood glucose monitor data used in experiments.

**Simulated**. Our primary analyses are performed on data generated using the UVA-Padova simulator [Man et al., 2014] via a publicly available implementation [Xie, 2018]. For ten simulated individuals (the "adult" patients modeled in the simulator), we generated approximately 150 days worth of data each, in 30 day roll-outs of the simulator. Carbohydrate values serve as $x$ values, while CGM values and insulin delivered as output by the simulator serve as **b** values. We use noise proportional to the true carbohydrate value, as studies on the accuracy of carb counting report errors relative to the total carbohydrates consumed [Brazeau et al., 2012]. Also based on Brazeau et al. [2012], we use a noise distribution with a negative bias, as the carbohydrates were found to be more-often under-reported than not. We therefore set $y = (1 + \mathcal{N}(-.25, .5))x$. We then cap $y$ below and above at 1 and 200 to keep values realistic. We consider additional noise distributions as sensitivity analyses. Bolus values were calculated based on the noisy carbohydrate values. See **Appendix A.1** for more details on data generation.

**Real**. This dataset includes both the OHIOT1DM 2018 and 2020 datasets, developed for the Knowledge Discovery in Healthcare Data Blood Glucose Level Predication Challenge [Marling and Bunescu, 2018b, 2020b]. The data pertain to 12 individuals, each with approximately 10,000 5-minute samples for training and 2,500 for testing. 12% of glucose values are missing, but we do not include windows with missing glucose values. We do not include windows with more than one carbohydrate measurement in our analysis. We sum carbohydrates to the first timepoint if they are less than 15 minutes apart to maximize the amount of usable data. We include only individuals with at least 100 training carbohydrate measurements, as fewer are not sufficient for learning a model. We note that ground truth carbohydrate measurements are not available for this dataset. Only potentially noisy patient-estimated values are reported.

#### 3.4.2.2 Baselines and Upper Bound

For all non-coteaching methods, we train two DAEs in parallel and report results on their averaged output for a fair comparison. We also note that all models receive the same auxiliary variables (blood glucose/ insulin) as input in an identical fashion. Overall time complexity is similar for all methods because there is only one back propagation per sample per-DAE.

• **CAE**: An upper performance bound. This model is an autoencoder trained with ground truth data, which we would expect to perform better than any method without access to ground truth data. In this oracle approach, $x$ values are substituted for $y$ values during training, but $y$ values are used during testing.

• **NAC**: Our first baseline. A DAE that treats the noisy data as clean which has been shown to perform well in low noise settings [Xu et al., 2020b].

• **NR2N**: Our second baseline is noisier2noise [Moran et al., 2019], which uses the known noise distribution to recover the clean signal. **NR2N** trains similarly to **NAC**, but at evaluation time a transform is used to recover the clean values (briefly, if the distribution of $N$ is known and we set $M = N$, the model should learn to recover half of the noise so the value used at evaluation is $2\hat{y} - z$).

• **SUP**: Our motivating setting can be re-framed as a supervised learning problem: predict $y$ (or $x$) values using $\mathbf{b}$ values as input. Depending on the noise distribution, it is possible that a model trained on noisy $y$ values could learn to predict the correct $x$, using similar logic to that found in Lehtinen et al. [2018]. We therefore use this supervised setting as a naive baseline. We simply input $\mathbf{b}$ to the same network used in the DAE setting and calculate loss as $(\hat{y} - y)^2$ during training, but here the model has no information regarding $y$ or $z$. As in the DAE setting, at test time we evaluate $(\tilde{y} - x)^2$.

• **SUPCT**: We apply co-teaching to the supervised setting (**SUP**), to ensure that performance gains observed are due to the combination of DAEs and co-teaching, and not co-teaching alone. Here, the model is tuned and trained identically to **N$^{+}$2N**, except the model does not receive $y$ or $z$ values.

#### 3.4.2.3 Implementation & Training Details

Each DAE is implemented as a 2-layer bidirectional LSTM with 100 hidden units. The LSTM model was chosen because it has been shown to perform well in blood glucose forecasting, which is a closely related challenge [Rubin-Falcone et al., 2020, Mirshekarian et al., 2019, Rabby et al., 2021].The final hidden state is passed to a fully connected layer with a single output. The output of the model is added to the input value corresponding to $y$, so that the network is tasked with learning the error term rather than a complete reconstruction. Because we only aim to correct a single carbohydrate ($y$) value but use a time-based model, we set one dimension of the model input to be $y$ for all timepoints. Multiple auxiliary $\mathbf{b}$ signals are input to the LSTM through separate channels: blood glucose values and bolus and basal insulin are included in this way. We also carry over bolus insulin values (which occur sparsely) to the end of the input window, to increase their impact on gradient calculations. We threshold the output of each DAE at 0, because carbohydrates (our $x$ and $y$) values cannot be negative. For each co-teaching method and **NR2N**, hyperparameters were selected based on tuning to a single individual adult#001. A small number of options was considered through a simple grid search. Once selected, these hyperparameters were used across all datasets. In tuning on adult#001, a ground truth signal was used for validation. This is a limitation, as such a signal is generally not available in real-world scenarios. However, the fact that the hyperparameters were not tuned to each individual, highlights the robustness of the approach. Tuning is described in **Appendix A.2**. At evaluation we report the result of the average correction learned by both networks when $y$ values are given as input (*i.e.*, where $\tilde{y}_i = DAE_i(y, \mathbf{b})$, we report $L(x, (\tilde{y}^1 + \tilde{y}^2)/2)$).

For sample selection during co-teaching, we use mean squared percentage error ($100\% \cdot ((\hat{y} - y)/y)^2$) to avoid eliminating all high-valued $y$ samples, as they are likely to have higher noise values. As a noise function during training, we use $z = (1 + \mathcal{N}(0, .5))Bern(.5)y$, *i.e.* we add random noise to half of the samples so that the model can learn to utilize noisy $z$ information, and zero-out the other half so that the model has to learn to distinguish zero from non-zero $y$ values based on $\mathbf{b}$ alone. See **Appendix A.3** for additional training details.

#### 3.4.2.4 Evaluation

**Simulated Data Metrics.** When ground truth carbohydrate values are available, we use MSE between the denoised carbohydrates and true carbohydrate values as our metric to report the remaining noise ($mean((x - \tilde{y})^2)$, where $\tilde{y} = (\tilde{y}^1 + \tilde{y}^2)/2$). The lower this value, the more noise has been removed. Although we assume that these data are not available at training time, we use them for evaluation. Since it can be difficult to interpret the meaning of a difference in MSE, we also consider a clinically motivated evaluation metric: *time in range.* Time in range is a measure of blood glucose management and varies with the accuracy of the carbohydrate measurements. The more accurate the carbohydrate estimates the more time an individual will spend 'in range.' Here, we run a simulation of the subject of interest with the default basal bolus controller using bolus values calculated from the updated carbohydrate values, and report the proportion of time in the simulation that each individual spent with blood glucose values between 70 and 180, or the euglycemic/healthy range. This metric serves to indicate the real-world impact each approach might have. For both metrics, 95% confidence intervals are calculated for each subject using 1,000 bootstrap re-samples, and the average $2.5^{th}$ and $97.5^{th}$ percentiles across subjects are reported.

**Sensitivity Analysis.** To evaluate our model under different noise assumptions, we repeat our analysis with multiple noise generation methods ($x \rightarrow y$), without altering hyperparameters or our $y \rightarrow z$ function. We use various Gaussian and uniform distributions, which include zero and negative mean multiplicative and additive noise functions. We consider noise functions that might arise in carbohydrate counting. None are highly dissimilar from our main analysis noise function: we aim here at feasibility, rather than a comprehensive survey on a broad selection of loss functions, which our method would likely be unable to address without further tuning or modification. Here, $\mathcal{U}(a, b)$ denotes a uniform distribution with values between $a$ and **b**. Carbohydrate values range between 0 and 200. After adding noise, $y$ values are capped above and below by 1 and 200. Alternate noise functions include:

1. Zero-mean multiplicative Gaussian: $y = (1 + \mathcal{N}(0, .75))x$
2. Negative-mean multiplicative Gaussian (primary noise function): $y = (1 + \mathcal{N}(-.25, .5))x$
3. Zero-mean additive Gaussian: $y = x + \mathcal{N}(0, 40)$

4. Negative-mean additive Gaussian: $y = x + \mathcal{N}(-30, 50)$
5. Zero-mean multiplicative Uniform: $y = \mathcal{U}(.5, 1.5)x$
6. Negative-mean multiplicative Uniform: $y = \mathcal{U}(0, 1.6)x$
7. Zero-mean additive Uniform: $y = x + \mathcal{U}(-60, 60)$
8. Negative-mean additive Uniform: $y = x + \mathcal{U}(-60, 40)$

**Real Data Analysis.** Without access to ground truth carbohydrate values at test time for the real dataset (unlike the simulated dataset), we evaluate the performance of our denoising approach based on a proxy. We take advantage of the fact that poorly estimated carbohydrates result in inappropriate bolus calculations, which result in poor blood glucose management. We expect that inaccurate carbohydrates estimates (large $[x - y]$ values) result in the poor blood glucose management. For a model that has come close to estimating $x$ correctly, we would observe a correlation between $[\tilde{y} - y]$ values and blood glucose control in the time period following a meal. We assess this with Correction-Risk-Correlation (CRC), defined as the Spearman correlation between the squared carbohydrate correction value ($(\tilde{y} - y)^2$) and the average Magni Risk [Magni et al., 2007] of blood glucose in the second hour following the carbohydrate. We use the Spearman correlation to account for non-linearities in the risk and correction value distributions. Magni risk is a measure of how far from a safe value blood glucose is; higher risk values correspond to blood glucose values that are either dangerously high or dangerously low. We use the second hour following the carbohydrate because the effects of the carbohydrate consumption and insulin bolus have not fully taken effect in the first hour. We calculate this correlation across all carbohydrates observed in all individuals. For validation purposes, we also calculate this metric for the simulated dataset.

Table 3.1: Our approach outperforms baselines for all evaluation metrics and both datasets, falling only 2% short of the upper bound for the clinical measurement 'Time in Range.' 95% CIs are calculated from 1,000 bootstrap re-samples. CRC $r$ and $p$ values are calculated from the Spearman correlation.

| | SIMULATED | | | REAL |
|---|---|---|---|---|
| Model | Remaining Carb MSE($g^2$)[95% CI] | Time in Range (%)[95% CI] | CRC ($r$)[$p$] | CRC ($r$)[$p$] |
| N/A-clean carb | 0.00 [0.00, 0.00] | 73.18 [72.49,73.88] | N/A | N/A |
| N/A-noisy carb | 72.26 [54.16, 92.58] | 65.43 [64.70,66.16] | N/A | N/A |
| CAE (Oracle) | 6.96 [5.03, 9.18] | 72.44 [71.76,73.11] | 0.32 [$< 0.001$] | N/A |
| SUP | 58.37 [45.11, 73.31] | 64.59 [63.89,65.30] | 0.04 [0.14] | 0.19 [$< 0.001$] |
| SUPCT | 100.50 [84.12,118.40] | 60.22 [59.43,60.94] | $< 0.001$ [0.91] | 0.03 [0.61] |
| NAC | 36.40 [27.02, 46.95] | 68.22 [67.53,68.93] | 0.11 [$< 0.001$] | 0.13 [0.05] |
| NR2N | 33.44 [26.59, 43.23] | 68.79 [68.09,69.51] | 0.11 [$< 0.001$] | 0.13 [0.05] |
| N$^+$2N (Ours) | **17.91 [12.61,24.98]** | **71.24 [70.54,71.90]** | **0.19 [$< 0.001$]** | **0.22 [$< 0.001$]** |

## 3.5 Results

Through our experiments, we aim to answer the following questions.

- Does our approach meaningfully reduce error across a variety of simulated individuals, compared to existing approaches?
- Is our model robust to different domain-appropriate noise distributions?
- Does our model show strong performance in real data, indicating accurate denoising?

**Error Reduction for Simulated Data.** Our approach, **N$^+$2N**, outperforms baselines in terms of noise reduction (MSE) (**Table 3.1**). **N$^+$2N** reduces MSE from $72g^2$ to $18g^2$, but falls short of the value achieved by our oracle approach **CAE** ($7g^2$), as expected. **CAE** does not achieve perfect MSE, probably due to insufficient training data or to a small amount of noise in the CGM signal. Our approach's reduction in noise is meaningful since it leads to significantly better time in range. Most methods offer an improvement in % time in range when used in a basal bolus controller, with baselines increasing over the noisy value from 65% to 69%, and **N$^+$2N** further improving performance to 71%, recovering 6% time in range out of a total of 8% lost when using noisy versus clean values. **SUPCT** performs worse than any other method including **SUP**, likely because without the noisy carbohydrate measurement as

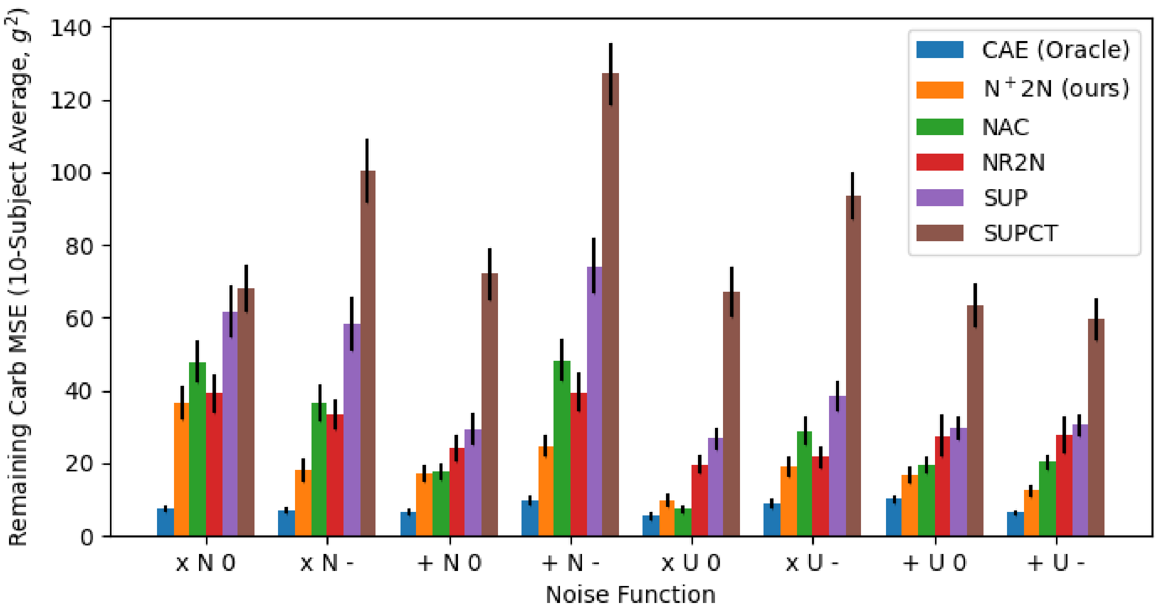


Figure 3.4: Performance on simulated datasets with multiplicative ($\times$) vs. additive ($+$), Normal ($N$) vs. Uniform ($U$), and zero (0) vs. negative mean ($-$) noise functions. **N$^+$2N** generally outperforms baselines. Error bars represent standard error (68% confidence interval) from 1000 bootstrap samples.

input, co-teaching cannot learn the relationship between $\mathbf{b}$ and $y$ as easily, and therefore does not identify the less corrupted samples during training. This results in essentially random sub-sample selection, hampering performance as less training data becomes available. **SUP** does not suffer from this problem because it always utilizes the entire dataset.

**Sensitivity Analyses on Simulated Data.** **N$^+$2N** outperforms all baselines across the majority of noise distributions (**Figure 3.4**). For zero-mean uniform multiplicative noise, **NAC** outperforms the proposed approach. We hypothesize that **NAC** performs well in this setting because the expected value of the noise is zero and the variance is lower than in other settings (it is 33%, which is approximately 30g, compared to 75% in the multiplicative normal setting, or 40g and 60g in the additive noise settings, see **Appendix A.4**). Of note, this analysis was carried out without additional tuning, demonstrating the resilience of our approach to varying noise assumptions. Across biased noised distributions, our proposed approach consistently outperforms *all* baselines. This resilience is likely due to our approach's ability to select clean samples for training, without reliance on the secondary noise function used during training.

We found that our approach is fairly robust to additional noise in the auxiliary signal: when we increased the amount of noise in **b** by 5X, our approach remained competitive with baselines trained with a clean **b** signal. At all noise settings our approach strongly outperformed baselines trained with similar data. Details and figures are in **Appendix A.5**.

In our examination of final training sample size, we found that performance is relatively stable with a larger, noisier, sample, with all $\tau$ values between zero and 0.5 (half of the samples excluded) offering substantial improvement over the best performing baseline. Performance degrades for larger values of $\tau$, which is likely due to the limited number of training samples. See **Appendix A.6** for details and a plot of model performance as $\tau$ is varied.

**Experiments on Real Data.** For the real dataset, **N$^+$2N** outperforms all baselines with respect to CRC. For simulated data, we see that, without exception, models with lower remaining MSE after denoising have a higher or equal CRC. This indicates that our metric serves as a reasonable proxy for remaining error when true values are unavailable. Plots showing the components used to calculate CRC (magnitude of carbohydrate correction versus Magni risk an hour after the meal) can be found in **Appendix A.7**. Interestingly, the **SUP** baseline performs fairly well for this task on the real dataset (r,p=0.19, 3e-3, vs. proposed approach 0.22, 9e-4). We hypothesize that this may be because carbohydrate measurements are so unreliable for this dataset that learning to predict them from scratch (without access to noisy values at test time) is sufficient for an error estimation proportional to the actual error, especially given the implicitly correct timing data.

## 3.6 Discussion & Conclusion

We propose a new approach to denoising, 'Noise$^+$2Noise', that does not assume access to ground truth target samples. Our approach leverages an auxiliary time-series that is related to the target signal to help identify target samples with less noise. Our approach is the first to adapt co-teaching to denoising, extending the applicability of this method to many potential settings. While the approach recovers target data retrospectively and cannot be used in real time for forecasting, it could be used in a number of downstream tasks. For example in the clinical context, errors in carbohydrate measurements could aid in evaluating an individual's efforts in blood glucose management and provide a potential target. In the context of carbohydrate recovery for blood glucose management, compared to existing approaches, 'Noise$^+$2Noise' leads to better signal reconstruction that is both statistically significant and clinically significant.

While promising, our approach is not without limitations. Our primary analyses are on simulated data where ground truth labels are available, but in real datasets common evaluation metrics (e.g., MSE) do not apply and we must rely on proxies. As presented, our approach is designed for retrospective carbohydrate correction; more work is necessary to investigate its applicability to closer-to-real-time correction. Four individuals in the Real dataset had too few carbohydrate measurements to reliably train a denoising model, which means that further work on model efficiency is necessary for this model to be broadly applicable. While our approach was designed for and evaluated on denoising non-missing measurements, our method could be extended to address missing measurements as well, provided some additional regularization is utilized to ensure that the model does not impute meal announcements too excessively. Finally, while we have empirically shown that co-teaching appears to select a low-noise sample, we have not provided statistical guarantees.

Our approach is developed under the assumption that low-noise samples span the entire input space. If this assumption is violated, e.g. if an individual consistently underestimates large meals, it may result in a distribution shift between the final training sample, which lacks representation in the excluded range, and the evaluation sample. Although such a discrepancy could impact performance, our method is somewhat robust to this scenario. This resilience is due to the utilization of all samples during the initial training phase, which provides some regularization that mitigates the effect of partial distribution exclusion later in training.

Despite these limitations, we have demonstrated that it is feasible to correct a noisy variable without access to ground truth samples during training, expanding the utility of ideas from image analysis and noisy label learning. Applied to domains in which data are composed of both individual-reported data and data measured from reliable sensors (e.g., mHealth), our approach could aid in improving the fidelity of patient-reported data.

# CHAPTER 4

# Forecasting with Sparse but Informative Variables

## 4.1 Introduction

In the previous chapter, we focused on learning alternate, cleaner feature representations for carbohydrate information to be used for blood glucose management. In this chapter, we shift focus to forecasting blood glucose.

In time-series forecasting, the future values of a target signal can depend on both intrinsic and extrinsic effects. Intrinsic effects are dynamics that depend only on the current and past values of the target signal. In contrast, extrinsic effects are dynamics that arise due to auxiliary variables. In many cases, the inclusion of such auxiliary signals as input to a forecasting model, in addition to the target signal, results in more accurate forecasts [Chakraborty et al., 1992, Rockwell and Davis, 2016]. However, in other settings, including auxiliary variables as input to a forecasting model produces little to no improvement in forecast accuracy, even when there is a known relationship between the additional variables and the target signal. This is particularly true in forecasting blood glucose: carbohydrates consumed and bolus insulin administered both have well-known effects on blood glucose, but their inclusion as inputs to forecasting models has not, in general, led to significant improvements in performance over models based on blood glucose alone [Rubin-Falcone et al., 2020, Hameed and Klienberg, 2020, McShinsky and Marsha, 2020]. We hypothesize

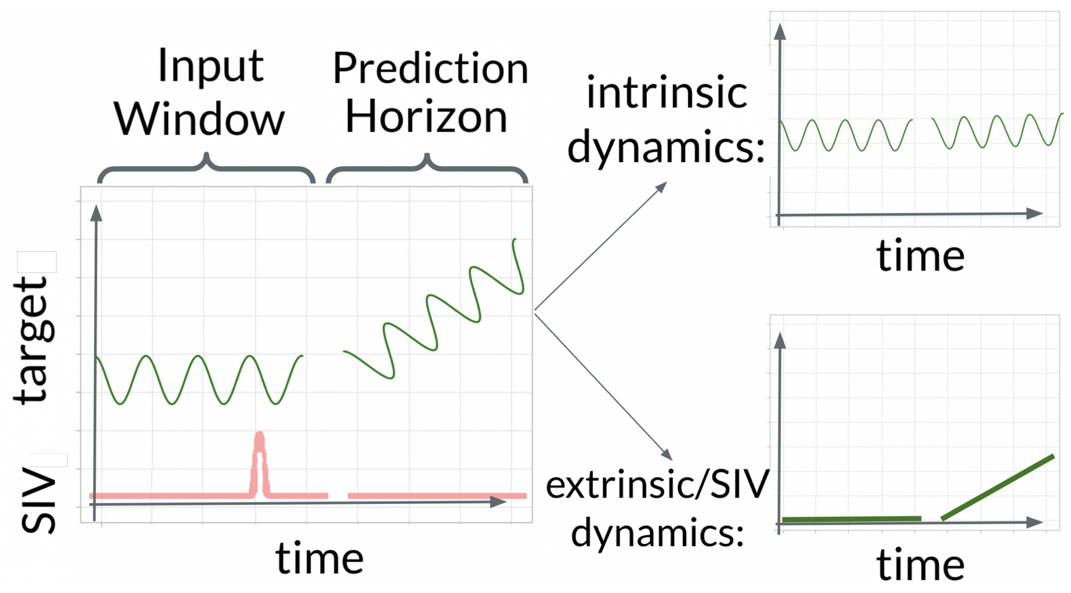


Figure 4.1: An overview of the SIV problem. In this toy example, the target variable exhibits oscillatory behavior when only zero SIV values are present (intrinsic dynamics), and the presence of a non-zero SIV value causes the target signal to increase linearly (extrinsic [SIV] dynamics).

that this is due in part to a mismatch in the relative frequency of non-zero values between the auxiliary signal and the target signal. We refer to forecasting tasks where an auxiliary signal is sparse but has a known effect on the target signal as the *sparse but informative variable (SIV)* problem.

In this chapter, we introduce and address the SIV problem (**Figure 4.1**), which arises when an auxiliary variable that occurs infrequently is known to cause an increase or decrease in the target variable's magnitude over time, although the exact effect may be unknown. The sparsity of the SIV often results in the failure of standard multi-input forecasting approaches in leveraging the auxiliary variable, *i.e.*, models that include the variable perform similarly to univariate-input approaches. A model that has overcome the SIV problem utilizes the SIV in making its predictions, resulting in improved forecasting accuracy relative to a model that does not use the additional variable. The SIV problem occurs when an important variable is mostly zero-valued. This is *not* the same as a sparsely *sampled* variable (SSV). In the case of under-sampling, the variable is sparse because it is not measured. In the SIV problem, we assume that the variable is measured frequently and accurately, but for most timepoints it is zero. While approaches for addressing missingness or SSVs have been extensively studied [Pratama et al., 2016], the SIV problem has not.

Although developing forecasting strategies that make use of SIVs has the potential to improve predictive accuracy in several domains, it has not been directly addressed in previous work. Recent multivariate forecasting approaches attempt to learn complex inter-variable dependencies [Qin et al., 2017, Freiburghaus et al., 2020, Xu et al., 2020a]. However, these approaches do not explicitly account for the relative sparsity of some variables. As we will demonstrate, the incorporation of domain knowledge in terms of restricting model outputs could help encourage a forecasting model to make better use of the SIV. However, existing state-of-the-art deep forecasting approaches generally do not use such restrictions in order to maintain flexibility. Here, we strive for a combination of the two: a forecasting approach that maintains flexibility while incorporating domain knowledge.

To address the SIV problem we propose a novel forecasting approach: "The Linked Encoder/Decoder." Our model integrates two main ideas: (i) the isolation of intrinsic and extrinsic effects, and (ii) the incorporation of domain knowledge. We implement the first idea with two separate but connected decoder networks. One network learns per-timepoint SIV effects (the SIV network), and the other learns the intrinsic dynamics of the target variable (the target network). We implement the second idea by restricting the output of the SIV network based on domain knowledge. Combined, these ideas lead to overall improved usage of the SIV and in turn more accurate forecasts.

Our main contributions are as follows:

- We present the sparse informative variable (SIV) problem.
- We propose a novel forecasting approach designed to leverage the SIV by isolating the effect of the SIV and incorporating domain knowledge.
- We evaluate our model in the context of forecasting blood glucose measurements and show that it more effectively incorporates SIVs compared to several baselines, even in the

presence of a small amount of noise.

## 4.2 Background and Related Work

We are the first to identify the SIV problem that arises when using RNNs for multi-input forecasting and the first to propose a solution. While sparsely sampled variables (SSVs) have been studied Pratama et al. [2016], the SIV problem is distinct. Interpolation approaches for addressing missingness and noise in SSVs are not directly applicable to the SIV setting. Although the SIV problem has not yet been addressed, several techniques have been proposed to learn inter-variable relationships in forecasting tasks, which in part inspire our approach. In the context of multi-input forecasting, Pantiskas et al. and Qin et al. use attention mechanisms to identify which variables to focus on Qin et al. [2017], Pantiskas et al. [2020]. However, in contrast to our approach, these approaches do not account for signals that are mostly zero-valued, nor do they incorporate domain knowledge. Beyond attention based mechanisms, in a probabilistic setting, normalizing flows have been used to directly model the joint probabilities between variables Rasul et al. [2020], Emmanuel de Bezenca an et al. [2020]. However, SIVs are often too sparse to accurately estimate a joint probability. Several other approaches have been proposed to explicitly model inter-variable relationships Gu et al. [2020], Freiburghaus et al. [2020], Pantiskas et al. [2020], Cao et al. [2020], Xu et al. [2020a]. However, none explicitly addresses the sparsity issue. Moreover, while some of these architectures separate the effects of variables, none use this isolation to restrict the effects based on domain knowledge as we do. There has been other work in forecasting that combines deep learning with domain knowledge to reduce the hypothesis space. However, researchers have relied on strong assumptions, *e.g.*, structuring deep architectures to match clinical intuition Munoz-Organero [2020], combining deep approaches with physiological-model-based simulators Miller et al. [2020], and estimating expert judgements on model outputs via Monte-Carlo approximations Huanga et al. [2014]. In contrast, we only restrict the sign of the SIV network's hidden state, a more flexible approach.

## 4.3 Problem Setup

We focus on the task of multi-input univariate-output time-series forecasting in which we aim to predict the future values of a single target variable, $x \in \mathbb{R}$, but have access to an additional auxiliary variable $x' \in \mathbb{R}$ that is sparse but informative. More specifically, $x'$ is zero at a much higher frequency than the target signal and the presence of a non-zero $x'$ value has a known effect on the target signal (*e.g.*, it results in either an increase or decrease). Given data pertaining to the previous $T$ values of the target signal, $\mathbf{x}_{-T+1:0}$, and the auxiliary signal $\mathbf{x}'_{-T+1:0}$, we aim to predict the next $h$ timepoints of the target signal: $\mathbf{y} = \mathbf{x}_{1:h}$. We denote the prediction output by a model as $\hat{\mathbf{y}}$

## 4.4 Methods

In this section, we present our proposed architecture for addressing the SIV problem, the "Linked Encoder/ Decoder". We then discuss our experimental setup.

### 4.4.1 Proposed Architecture

**Overview.** To effectively capture the autoregressive dynamics of forecasting with an SIV, our architecture, the "Linked Encoder/ Decoder," relies on a recursive framework (**Figure 4.2**). It involves one encoder network and two linked decoder networks, which are used to isolate the SIV dynamics from the intrinsic dynamics. The SIV decoder receives the SIV signal as input. The decoder systems share a hidden state, which is processed in parallel. This allows the intrinsic and extrinsic dynamics to be learned separately. Once isolated, we restrict the effect of the SIV on the target signal based on domain knowledge.

**Standard Encoder/Decoder.** Our approach is based on a standard encoder/decoder

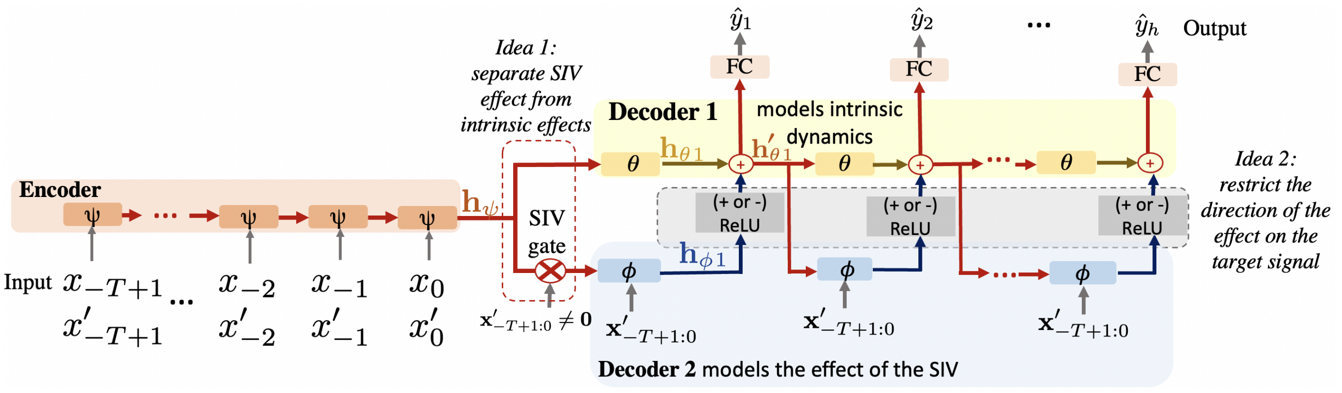

Idea 1: separate SIV effect from intrinsic effects
Encoder
Input
Decoder 1
models intrinsic dynamics
SIV gate
FC
Output
(+ or -) ReLU
Idea 2: restrict the direction of the effect on the target signal
Decoder 2 models the effect of the SIV

Figure 4.2: Our architecture: the Linked Encoder/Decoder, shown with an input length $T$ and prediction horizon $h$. The $\theta$ network models intrinsic dynamics, while the $\phi$ network models the SIV dynamics. The $\psi$ network models shared dynamics. Input time-series are gated, such that only inputs ($[\mathbf{x}_{-T+1:0}, \mathbf{x}'_{-T+1:0}]$) containing a non-zero SIV at any timepoint are passed through the $\phi$ network. The ReLU network shown in grey is used to ensure that the relationship between SIV and target is as expected.

recurrent neural network, as depicted by the orange and yellow sections of **Figure 4.2**. For samples where $\mathbf{x}'_{-T+1:0} = \mathbf{0}$, this encoder/decoder is not modified. A single encoder ($\psi$), takes $\mathbf{x}_{-T+1:0}$ and $\mathbf{x}'_{-T+1:0}$ as input. The encoder outputs a hidden state $\mathbf{h}_\psi = \psi([\mathbf{x}_{-T+1:0}; \mathbf{x}'_{-T+1:0}])$, which is passed through a decoder LSTM ($\theta$) that outputs hidden state $\mathbf{h}_{\theta 1} = \theta(\mathbf{h}_\psi)$. At each timepoint in the forecast horizon, the output from the previous time step $\mathbf{h}_{\theta t-1}$ is passed through $\theta$, such that $\mathbf{h}_{\theta t} = \theta(\mathbf{h}_{\theta t-1})$. This learned representation is also passed through fully connected output network $FC$ at each time step $t$ in the forecast horizon to output a prediction $\hat{y}_t = FC(\mathbf{h}_{\theta t})$.

**Linked SIV Decoder and Gating.** If an input sample contains a non-zero SIV value, it is passed through both this standard encoder/decoder and a second decoder $\phi$, depicted in the blue section of **Figure 4.2**. This second decoder aims to model SIV dynamics. By gating samples based on SIV values, we separate the extrinsic effects of the SIV on the target variable from the intrinsic effects of the target signal on itself. When the corresponding SIV values are non-zero (*i.e.*, $\mathbf{x}'_{-T+1:0} \neq \mathbf{0}$), the network engages the second decoder, which processes hidden state $\mathbf{h}_{\theta t}$ for $t = 1, ..., h$, in parallel with $\theta$, as described below. Because $\phi$ is only engaged when an SIV is present, $\theta$ learns to forecast in the absence of an SIV, while

$\phi$ learns the additional SIV effect.

**Linked Hidden State Processing and Incorporating Domain Knowledge.** Both decoders process a single hidden state in parallel, and their outputs are summed. At the first time step in the prediction horizon, both decoders take as input $\mathbf{h}_{\psi}$, but they each output a unique hidden state ($\mathbf{h}_{\theta t}$ and $\mathbf{h}_{\phi_t}$). At subsequent time steps, these two hidden states are summed to create a new hidden state: as in Figure 4.2, we define $\mathbf{h}'_{\theta t} = \mathbf{h}_{\theta t} + k \cdot max(\mathbf{h}_{\phi t}, \mathbf{0})$, where $k = 1$ if the SIV is known to lead to an increase in the target signal and $k = -1$ otherwise. This is equivilent to passing $\mathbf{h}_{\phi t}$ through a ReLU function, which restricts the effect of the SIV on the target variable to the expected direction. The combined hidden state ($\mathbf{h}'_{\theta t}$) is passed to the output network ($FC$) and both decoders at subsequent time steps. The final forecast $\hat{\mathbf{y}}$ is a product of both decoders, capturing both intrinsic and restricted extrinsic effects (*i.e.*, $\hat{y}_t = FC(\mathbf{h}'_{\theta t})$). Note that when $\mathbf{x}' = \mathbf{0}$, $\phi$ is not engaged, and $\mathbf{h}'_{\theta t} = \mathbf{h}_{\theta t}$. In order to encourage the SIV decoder ($\phi$) to utilize the SIV, $\phi$ receives the entire SIV signal as input, concatenated with the hidden state. The SIV signal is shifted at each timepoint so that the encoder's position in time relative to the SIV is included in the representation implicitly (see implementation details).

**Additional Variables.** In **Figure 4.2**, we present an overview of the proposed architecture for a setting with a single SIV. In a setting with multiple SIVs, the number of secondary decoders is increased and restrictions are applied according to the known effect of each SIV. Each $\phi$ takes as input only the relevant SIV signal, along with $\mathbf{h}'_{\theta t}$. $\mathbf{h}'_{\theta t}$ is modified by all SIV decoder systems, so that the hidden state that is passed to $FC$ and subsequent decoder steps is a sum of the number of SIVs plus one components. Non-sparse auxiliary variables, if any, are given to the $\psi$ network.

**SIV Representation.** One issue that makes utilizing SIVs difficult is that they usually occur at only one timepoint in an input window, having little effect on the gradient during training. To increase its effect, we use the sum-total SIV value up to the current timepoint as input (**Figure 4.3**). Up until the first non-zero SIV value of an input time-series, the signal

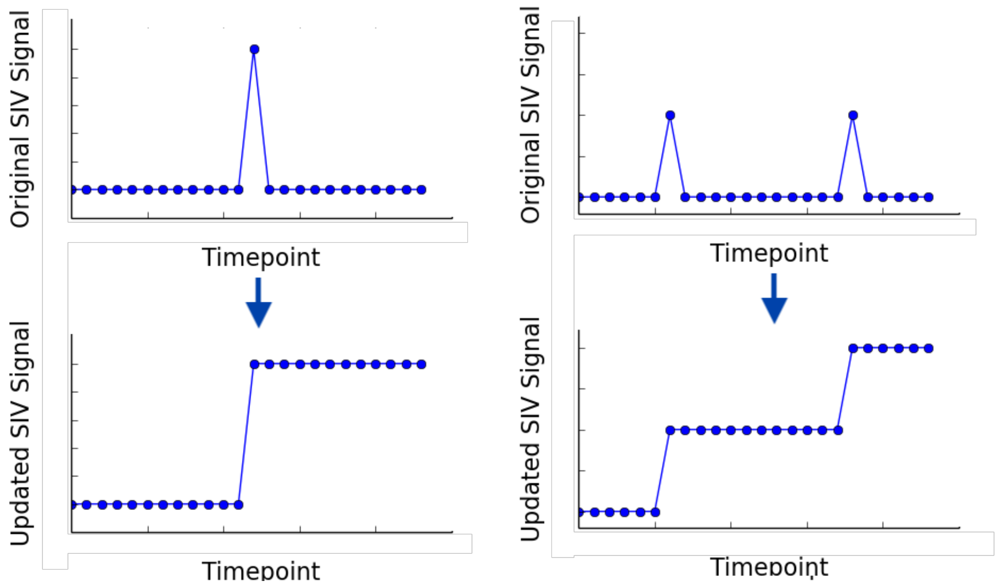


Figure 4.3: Our sum-total approach. We use the sum total up to the current point within an input window as input. This method allows the SIV signal to make a larger impact on the gradient while maintaining all temporal information.

value is zero. After any non-zero SIV, the input is the sum of observed values prior to and including that point, *in the input time-series only*. SIV values from before the input window are ignored. This approach is used for all analyses, including baselines and ablations. This sum-total input improved performance for all approaches (see **Appendix B.2**).

**Implementation Details.** Each LSTM encoder or decoder is implemented as a 2-layer bidirectional LSTM with 100 hidden units. $FC$ is a fully connected linear network with a single output. Our architecture uses two $\phi$ networks, one for carbohydrates (positive effect, ReLU restriction) and one for bolus insulin (negative effect, $-1\times$ ReLU restriction). In order to input each SIV signal into each $\phi$ network while maintaining time information, $\mathbf{x}'_{-T+1:0}$ is front-padded with $h$ zeros and input to $\phi$ at the first time step of the forecast horizon. The signal is shifted back at each timepoint, such that at the $i^{th}$ time step of the prediction horizon, the SIV signal is shifted back $i-1$ positions, so that it is front-padded with $h-(i-1)$ zeros and back-padded with $i-1$ zeros. In this way the input corresponds with the encoder's position in time. $\mathbf{x}'_{-T+1:0}$ is scaled to have the same mean as $\mathbf{h}'_{\theta t}$ for each input.

### 4.4.2 Evaluating Model Performance

We evaluate our approach on several datasets pertaining to BG forecasting in type 1 diabetes. We compare to several baselines, including resampling approaches. We evaluate forecast accuracy and the extent to which each model utilizes the SIVs.

We aim to forecast BG 30 minutes into the future ($h = 6$), based on a history of BG and two SIVs: carbohydrate and insulin bolus values. We use an input length of 2 hours ($T = 24$). Here, $h = 6$ as it represents a common BG forecasting benchmark [Marling and Bunescu, 2018a] and we set T=24 based on prior work that suggests longer histories do not provide additional benefit [Silvia Oviedo, 2016]. Target and auxiliary variables are scaled to between zero and one, using the maximum expected value to linearly scale (400 for BG, 200 for carbohydrates, 50 for insulin). Each individual's train/validation/test data are split into overlapping windows of length $T + h$ with a stride of 1, to be used as model input and labels.

#### 4.4.2.1 Data

We compare the performance of our architecture to baselines on three T1D-based datasets: Simulated, Simulated-noisy, and Ohio. All three datasets are publicly available [Man et al., 2014, Xie, 2018, Marling and Bunescu, 2018a].

*Simulated.* Data generated from a commonly-used type 1 diabetes simulator [Man et al., 2014, Xie, 2018] provide a curated test setting on which to evaluate our approach. We generate ten days of data for ten individuals corresponding to 28,800 timepoints.

*Simulated - noisy.* While the simulator used above introduces noise in the BG measurements, we assume that the SIV signals are recorded without noise or missingness. To measure the robustness of our approach to this assumption, we generate additional simulated datasets using the approach above in which we vary the amount of noise and missingness. After data generation, we zero out between 10% and 50% of the carbohydrate values. Separately, we also explore the effects of adding uniform random noise to the measurements, varying the maximum magnitude between 10% and 50%. These changes are made to both training and testing data.

*Ohio.* Finally, we examine performance on a real dataset that was made publicly available for the Knowledge Discovery in Healthcare Data BG Level Predication Challenge [Marling and Bunescu, 2018a]. The data pertain to 12 individuals with BG measurements every 5 minutes.

#### 4.4.2.2 Baselines

**Encoder/Decoder** (Abbrev: Enc/Dec). Our primary baseline is a stand-alone encoder/ decoder system, identical to the $\psi$ plus $\theta$ setup in our full architecture [Fox et al., 2018]. This baseline has less capacity than our proposed approach, so we increase the minimum number of training iterations to match the same number of gradient updates for our approach. We also examine a model with capacity that matches our proposed approach (Full Capacity).

**Full Capacity.** To ensure that any performance improvements observed are not due to our model's increased capacity, we also compare to a model based on our full architecture, but with no SIV-specialization (*i.e.*, there is no gating, no direct SIV input into $\phi$, and no output restriction). This model is trained for the same number of iterations as our proposed approach.

**Resampling.** Resampling is perhaps the simplest approach to addressing signal sparsity. In order to rule out resampling as a naive solution to the SIV problem, we implement our primary baseline method with two resampling procedures: training the model on only windows with SIV samples to initialize the weights before training on the full sample (**SIV Initialize**), and training on the full sample, then fine-tuning the model on only windows with non-zero SIV values (**SIV Fine-tune**). Similar to our primary encoder/decoder baseline, we increase the minimum number of training iterations to match the number of gradient updates used in training our proposed method.

#### 4.4.2.3 Evaluation

All evaluations are performed on held-out test sets. To evaluate the accuracy of each model, we predominantly use root mean square error (rMSE; $\sqrt{(\frac{1}{n})\sum_{i=1}^{n}(y_i - \hat{y}i)^2}$) and mean absolute error (MAE; $(\frac{1}{n})\sum_{i=1}^{n}|y_i - \hat{y}_i|$). In order to match common practice in the BG forecasting literature, we calculate error terms based on the prediction accuracy of the final timepoint in the prediction window [Marling and Bunescu, 2018a]. We also evaluate the impact of carbohydrates and insulin on BG with a shapley value-inspired [Lipovetsky and Conklin, 2001] metric called SIV Usage. This metric is calculated as follows. Let $\mathbb{X}$ denote a dataset with an SIV, and let $\mathbb{X}_\emptyset$ denote the exact same dataset with all SIV values set to zero. Let $f$ define a mapping $f : X \rightarrow \hat{\mathbf{y}}$, where $X \in \mathbb{X}$, and $\hat{\mathbf{y}} \in \mathbb{R}^h$ is a prediction of the next $h$ points of the target variable. $f$ is trained and evaluated on $\mathbb{X}$, while $f_\emptyset$ is trained and evaluated identically to $f$, except using $\mathbb{X}_\emptyset$. Let $L$ denote the error of the model's prediction (here, rMSE or MAE). We define SIV usage as $L(f_\emptyset(\mathbb{X}_\emptyset))$ - $L(f(\mathbb{X}))$. This metric reflects how error changes when the SIV are removed. When removing the SIV, we both train and test on data without the SIV (rather than performing a permutation test or similar), so the model can learn the maximum amount of information available from the target variable alone.

**Individual-Level Analyses.** We compare the error and SIV usage of the baseline encoder/decoder to the improvement over baseline offered by our method across individuals. We expect that our model will offer greater improvement over baseline for individuals with high baseline error and low baseline SIV usage, as those are the individuals for which SIVs are most poorly modeled in the baseline. This would indicate that our approach addresses baseline deficits in SIV-modelling.

**Ablations.** We perform the following ablation analyses to examine which elements contribute to our model's performance. **No Gating**: All samples are passed through both decoders. **No Restriction**: The outputs of the SIV decoder systems are not passed through a ReLU function. **No SIV Input**: The SIV decoders receive only the hidden state as input, and not the SIV signal. **Only SIV Input**: The baseline model is used, but with the modification that the SIV is input to the decoder directly, as in our proposed approach.

**Noise and Missingness.** We assume that SIV signals are noise free. Here, we evaluate our model's performance as this assumption is relaxed. This is possible in our simulated dataset because we have ground truth carbohydrate values. In real data, not only may the magnitude of these values be inaccurate (they are estimated by the individual), but individuals may skip recordings altogether. In order to examine how unreliable carbohydrate values impact our approach, we utilize the varied-noise and varied-missingness simulated-noisy datasets to evaluate the performance of our model vs the stand-alone encoder/decoder baseline as carbohydrate values become unreliable. We report average forecast error across all individuals and 5 random noise-generation seeds for this analysis. We also evaluate our approach on a real dataset which is expected to have some degree of missingness and noise for comparison.

**Training Details.** We split each individual's data into training, validation and test sets. We split data by number of timepoints using a 70%/15%/15% split, without overlap. We implement and train our models in pytorch 1.9.1 and CUDA version 10.2, using Ubuntu 16.04.7 and a GeForce RTX 2080, using an Adam optimizer Kingma and Ba [2014] and a batch size of 500. We use a learning rate of 0.01 and a weight decay of $10^{-7}$. When training, mean square error (MSE) across all timepoints in the prediction horizon is used as a loss function. We train for at least 500 epochs, until validation performance does not improve for 50 epochs. The model parameters that led to the best validation set performance are used at inference time. We train and test a model on each individual and report across-individual averages.

## 4.5 Results

We aim to answer the following questions.

- Does our model offer improvement over baseline approaches in terms of forecast error

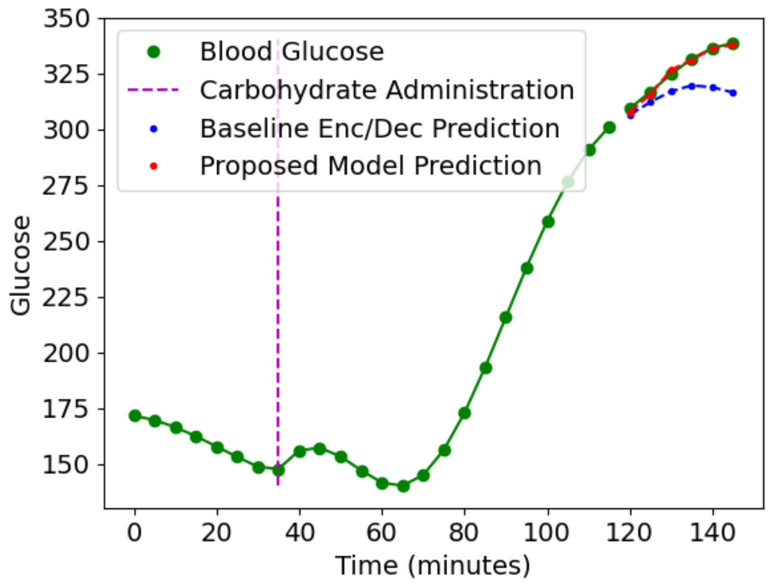


Figure 4.4: A sample prediction for a simulated individual (adult#004). Our model better accounts for the steep rise in the BG signal following a meal.

and SIV Usage?

- Across individuals, when does our model offer the greatest improvement?
- What elements of our model improve performance?
- How do inaccurately measured SIVs impact our model?

**Improvement Over Baselines (Simulated Data).** Our model outperforms baselines leading to lower average rMSE/ MAE and greater relative SIV usage (**Table 5.6**). Specifically, our model appears to account for the effect of the SIV on the target signal (*e.g.*, accurately predicting sharp rises that the baseline encoder/decoder cannot account for: **Figure 4.4**). Naive SIV resampling approaches are generally outperformed by other baseline approaches, or perform similarly. Our proposed approach shows a large improvement over even the strongest baseline (rMSE 13.07 vs 14.14, paired t-test across individuals [t statistic/p-value]: 2.0/0.05). We also examine the clinical benefit of forecasting with our approach using a Clarke error grid in **Appendix B.3**.

**Individual Level Results (Simulated Data).** In the simulated dataset, we consider ten individuals who differ in terms of simulated physiological parameters. Here, we inves-

| Model | rMSE [95%CI](Usage) | MAE [95%CI](Usage) |
|---|---|---|
| **BASELINES** | | |
| Enc/Dec | 15.63[14.1,16.9](11.13) | 12.42[11.1,13.6] (6.63) |
| SIV Fine-tune | 27.30[24.7,29.1](-0.54) | 22.22[19.9,23.9](-3.17) |
| SIV Initialize | 15.37[13.6,16.9](11.39) | 11.99[10.7,13.2] (7.06) |
| Full Capacity | 14.14[12.7,15.3](12.62) | 11.21 [9.9,12.2] (7.85) |
| **PROPOSED** | 13.07[11.8,14.2](13.69) | 10.45 [9.4,11.4] (8.61) |
| **ABLATIONS** | | |
| No Gating | 13.93[12.6,15.0](12.84) | 11.11 [9.9,12.1] (7.95) |
| No Restriction | 13.12[11.8,14.2](13.64) | 10.43 [9.3,11.4] (8.62) |
| No SIV Input | 14.20[12.7,15.4](12.56) | 11.18 [9.9,12.2] (7.87) |
| Only SIV Input | 13.97[12.6,15.1](12.79) | 11.12 [9.6,12.2] (7.94) |

Table 4.1: Mean forecasting error and SIV usage. Outcomes are reported as: Error [95% confidence interval] (SIV Usage). Our proposed approach outperforms baselines and ablations. CIs were calculated using 1,000 bootstrap samples.

tigate how model performance with respect to the baseline Encoder/Decoder varies across these individuals. In particular, we identify settings in which our approach is most beneficial.

Our model's benefit over baseline varies inversely with the extent to which the baseline approach relies on the SIV (*i.e.*, SIV usage) across individuals (Pearson r=-0.65, p=0.041, **Figure 4.5 (a)**). This supports the hypothesis that our model's improved performance over the baseline is due in part to the increased usage of the SIV. For individuals for whom the baseline model was able to achieve high usage, our model was not necessary, but individuals with low baseline usage stood to benefit. We also observe a strong correlation between baseline forecast error and our approach's improvement (r=0.80, p= 0.0056, **Figure 4.5 (b)**). This suggests that our approach addresses the deficits of the baseline at the individual level. There is higher rMSE variability across individuals for the baseline compared to the proposed approach (range: 9.5 vs 6.3), perhaps partially due to difficulties in SIV modeling, for which our model compensates.

**Ablations (Simulated data).** Ablation analyses reveal that, in general, our approach's strong SIV usage and forecast accuracy are a result of the combined effects of each implementation detail, rather than any one component. **Table 5.6** shows the results of our ablation

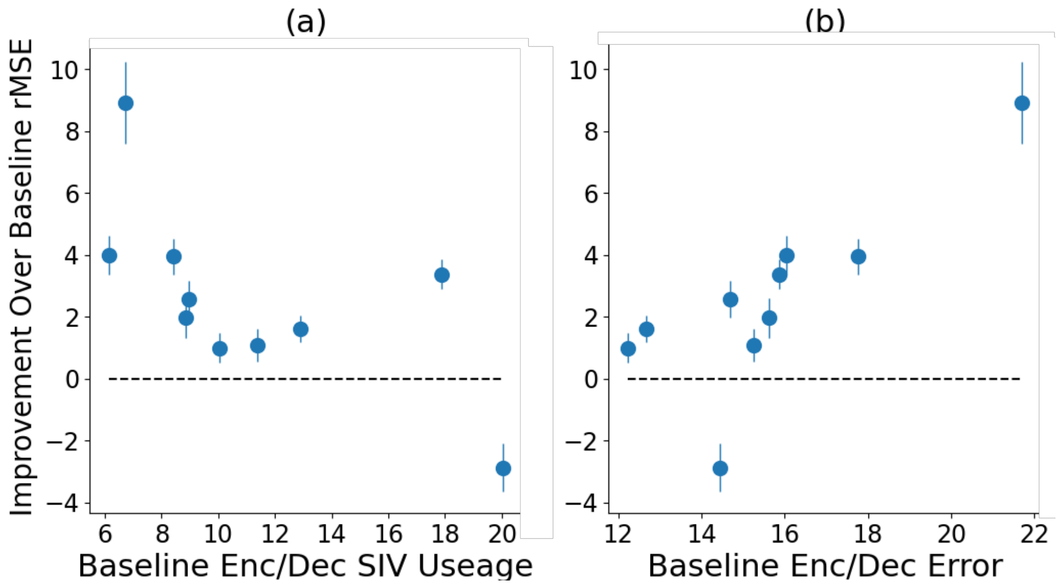


Figure 4.5: (a) Our architecture's improvement over baseline increases as baseline SIV usage decreases. (b) Improvement over baseline is positively correlated with baseline rMSE. Each point represents an individual in the simulated dataset, and errors bars represent standard error *i.e.*, the standard deviation of 100 bootstrapped samples.

study: removing any component results in a decrease in performance accuracy and SIV usage. Notably, removing the domain-guided restriction (*i.e.,* the ReLU functions) results in the smallest effect on performance. This is likely because, in our simulated dataset, the effect of the insulin boluses and carbohydrate administrations are strong enough that the model can learn them easily without supervision. When the restriction element is included, it seems to be utilized by the model: randomly switching the sign of the ReLU component at test time in a post-hoc analysis drastically hampers performance, more than doubling rMSE. We expect this component to have a greater effect in situations where the impact of the SIV is more subtle.

**Noise and Missingness (Simulated and Real data).** In the above experiments, we assume that the SIV is accurately measured. To quantify the impact of measurement noise on our approach, we perturb the SIV as described in the experimental setup section. We find that as missingness and noise increase, our approach's performance degrades (**Figure 4.6**). Relative to the baseline, our approach is more impacted by corrupted SIV values, in

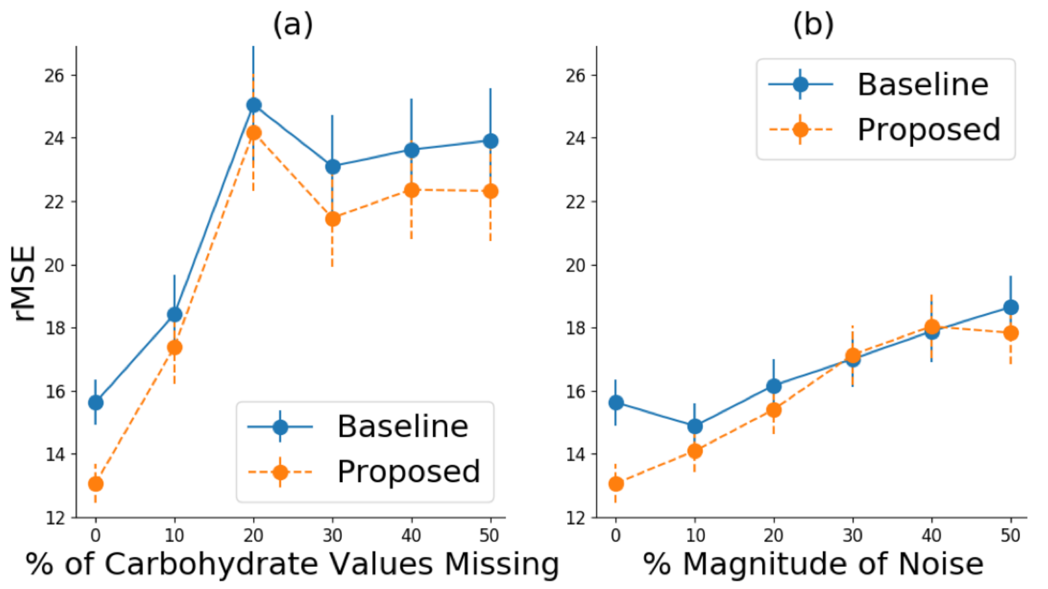


Figure 4.6: Mean model performance across all individuals as (a) simulated carbohydrate values are hidden and (b) noise is added to carbohydrates. As noise and missingness increase, our model fails to reliably outperform the baseline. Errors bars represent standard error *i.e.*, the standard deviation of 100 bootstrapped samples.

part because of the increased dependence on the SIV (*i.e.*, greater SIV usage). As expected, completely omitting carbohydrates has a greater effect than simply corrupting the magnitude. Though performance decreases with increased noise/missingness, we are encouraged by the fact that our proposed approach remains competitive with the baseline.

We further explore the effects of a corrupted SIV signal using the 'Ohio' data, a real dataset generated by individuals with type 1 diabetes. While we cannot measure the amount of noise in the Ohio dataset, we expect it to be more in line with the simulated-noise setting than the noise-free setting. Somewhat reassuringly, even in this noisy setting, our approach performs no worse than existing approaches and even provides a small benefit over baselines. Specifically, our approach consistently leads to lower forecast rMSE compared to all baselines, though performance gains are modest (rMSE=20.16 vs. the strongest baseline rMSE=20.36). Furthermore, in ablation analyses we found that the restriction component was beneficial for this dataset, supporting the hypothesis that domain knowledge insertion is beneficial for more challenging tasks (see **Appendix B.1** for full results).

## 4.6 Discussion & Conclusion

The SIV problem arises in forecasting domains when the relative sparsity of an auxiliary signal makes learning its effect on a target signal challenging. We introduce the problem and propose a forecasting approach that leverages SIVs. Our approach isolates SIV dynamics and restricts them based on domain knowledge, achieving higher SIV-usage and stronger forecasting performance than baselines. While our approach assumes accurately measured SIVs, it performs no worse than baselines in the presence of missing or noisy SIV measurements. Though we focus on a specific use case, we expect the SIV problem to arise frequently in healthcare. In such settings, SIVs are likely associated with time periods during which a patient is most vulnerable (*i.e.*, medication administration). Therefore, prediction models that address the SIV problem could lead to more accurate predictions during time periods that are critical for health outcomes.

While there are many different ways to forecast signals, we focus on RNN-based techniques. Our primary contributions are the identification of the SIV problem in forecasting and noting the failure of common RNN-based approaches when addressing this problem. We demonstrate how addressing the SIV problem can lead to improvements over directly comparable baselines. We do not claim SOTA in forecasting, but our findings could apply to many settings in which variants of RNNs are applied to forecasting problems with SIVs, which could include vital sign forecasting with medication administration as an SIV, stock prices with quarterly reports or news articles as SIVs, and traffic forecasting with holidays or events as SIVs. The two main ideas behind our approach include gating and output restriction. While neither of these methodological developments are unprecedented on their own, their combined application to the SIV problem poses a novel direction for forecasting in related domains.

CHAPTER 5

# Learning Control-Ready Forecasters for Blood Glucose Management

## 5.1 Introduction

Individuals with T1D must inject insulin to manage their blood glucose (BG) levels. Manual dosing solutions, such as the commonly used basal bolus (BB) regimen, require patient-managed parameters [Janez et al., 2020, Weinstock, 2023] that must be manually updated as an individual's glucoregulatory system evolves over time [Lee et al., 2021, Rubin-Falcone et al., 2022]. Even so-called "automated" systems rely heavily on patient-provided parameters, which demand calibration and oversight as an individual's body changes [Ahmed et al., 2020, Collyns et al., 2021, Ware et al., 2022].

An adaptive BG control system could alleviate the need for constant patient intervention. Along these lines, researchers have investigated machine learning (ML)-based forecasters for model-based control. Such forecasters can be continually retrained on the most recent data [Anava et al., 2013], adapting to an individual's changing glucoregulatory system without manual intervention. However, such forecasters often fail to accurately model BG across a range of potential insulin doses. In addition to challenges around sparsity addressed in Chapter 4, the inherent correlation between carbohydrate intake and bolus insulin values in BB-controlled data makes training effective forecasters difficult. The correlation leads to *entanglement* in which the forecaster fails to accurately model the independent impact of

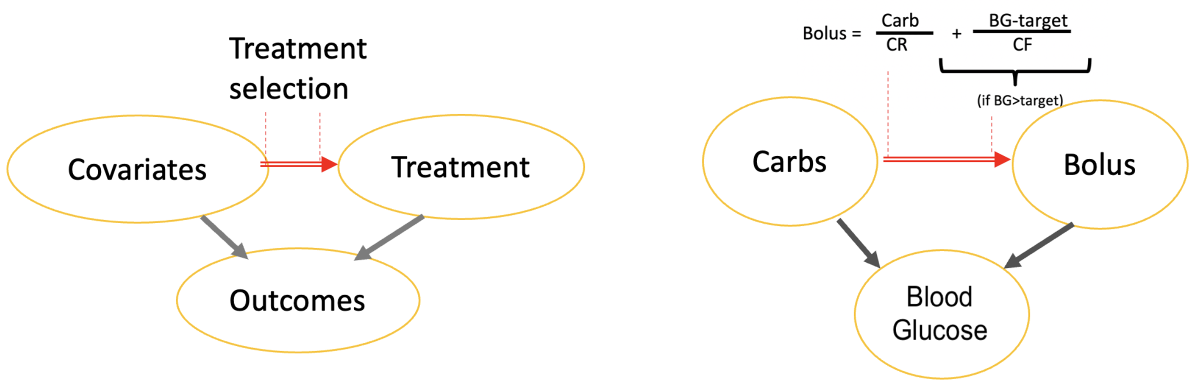


Figure 5.1: Our setting is similar to modeling treatment effects in the presence of confounding: the basal bolus strategy creates a strong correlation between bolus and carbohydrate values, making it difficult to disentangle their impacts on future BG values. Bolus values depend on three patient-specific parameters: BG target, carbohydrate ratio (CR), and correction factor (CF).

these variables.

Existing solutions, like recurrent neural network (RNN)-based forecasters trained on data collected under a BB policy, fail to address the entanglement present in BB data, since they do not adjust for the presence of confounding: the number of carbohydrates in a meal affect both the outcome (future BG values) and the treatment (insulin administration; see **Figure 5.1** for causal DAG). Treatment effect estimation techniques that adjust for confounding via matching or stratifying (such as those utilizing a propensity score [Collyns et al., 2021]), do not apply because the BB strategy permits no overlap; bolus choices are selected deterministically based on other system variables with no variability. While there is some overlap across individuals (i.e., two people eating similarly sized meals may use different bolus values), the significant heterogeneity in BG dynamics among individuals [Redondo and Morgan, 2023] prevents attributing differences in BG trajectories solely to bolus effects. Variations in predicted BG values between two individuals can be ascribed to a multitude of other factors, so a forecaster trained on data from multiple individuals cannot accurately model bolus effects specific to individuals [Huang et al., 2023]. In terms of patient specific models, disentangled representation learning techniques [Kingma and Welling, 2014, Jeon et al., 2021] do not account for the near-perfect correlation between action choices. Our domain knowledge utilization approach, the linked encoder/decoder technique described in Chapter 4, is able to address the orthogonal problem of auxiliary variable sparsity. However, approaches of this type cannot directly address entanglement without precise assumptions regarding the relative magnitudes and shapes of entangled variable impacts.

In this chapter, we propose a novel approach for training control-ready BG forecasters with the goal of learning to accurately model the independent effects of carbohydrates and insulin on BG. Our strategy hinges on the insight that depending on the BG value, the entanglement between boluses and carbohydrates can vary. Specifically, in the basal bolus (BB) regimen, the bolus calculation involves a corrective dose when BG levels are high. Based on this observation, we propose a two stage approach. In stage 1, we utilize samples with a correction factor to construct an initial estimate of the independent effect of insulin. In stage 2, we exploit the monotonic impact that insulin has on BG during subsequent training. This enables the forecaster to accurately disentangle action effects, making it ready for model-based control.

In simulations, our control system outperforms that based on a forecaster trained using standard approaches. Notably, our method does not require a user to tune any hyperparameters, enhancing its reliability over BB and other approaches. For instance, we show that as the parameters for the BB control strategy become inaccurate in a simulated setting, control performance degrades. This demonstrates a lack of adaptability to evolving system dynamics, which often arise in individuals with T1D, as an individual's body will change over time [Majety, 2022]. Our approach's inherent adaptability could potentially obviate the patient burden associated parameter updates as an individual's body changes. We further illustrate our approach's benefit over baseline forecasters using real-world data from three distinct datasets via proxy metrics which assess potential model-based control performance. Though inspired by BG management in T1D, our methodology has broader implications. It can be applied to other control settings, especially when behavior policies result in correlated action pairs, given that this correlation is not absolute and that one action's impact on the change in state is monotonic. We demonstrate this broader applicability in a classic inverted pendulum setting, in which we learn to forecast pendulum angle, accurately disentangling the effects of the forces of the agent's push and gravity. Our method has potential in safety-critical domains where exploration is expensive.

Our main contributions are as follows:

- We formalize the problem of learning the independent effects of insulin and carbohydrates on BG with access to only highly entangled BB data.
- We propose a novel approach in training control-ready BG forecasters that leverages correction bolus values and exploits insulin's monotonic impact on BG to learn an estimated effect for further training.
- Using an FDA-approved simulator, we demonstrate our approach's efficacy within a model-based control strategy, demonstrating improved control over baselines from causal inference, domain knowledge utilization, and disentangled representation learning.
- On real data from three different cohorts, we design and validate proxy metrics, demonstrating the efficacy of our proposed approach in a real-world setting.

Though inspired by BG management in T1D, our methodology has broader implications. It can be applied to other control settings, especially when behavior policies result in correlated action pairs, given that this correlation is not absolute and that one action's impact on the change in state is monotonic. Our method has potential in safety-critical domains where exploration is expensive.

## 5.2 Background and Related Work

In this section, we first describe the challenges and benefits of BG forecasting. Following this, we discuss model-based control strategies, for which our approach is specifically designed. Lastly, we discuss the limitations of existing treatment effect estimation techniques applied to the problem of learning control-ready forecasters.

### 5.2.1 BG Forecasting

In BG forecasting, one typically aims to forecast the next 15 minutes to two hours of BG measurements. Such forecasts can inform control strategies [Loop Docs, 2023, Yamagata et al., 2020]. However, the effectiveness of these forecasters, especially in guiding bolus dosing, is compromised if they fail to differentiate the individual effects of carbohydrates and insulin on BG. In model-based control scenarios, an inaccurate representation of carbohydrate and bolus impacts can lead to erroneous BG trajectory predictions and, consequently, inappropriate bolus dosing recommendations. Such inaccuracies can arise due to *distribution shift*: during control, forecasters are likely to encounter bolus, carbohydrate, and BG data combinations that differ substantially from those encountered by the behavior policy used for training. The behavior policy typically used to collect training data relies on a BB strategy, where insulin dosing depends on carbohydrate intake. Specifically, bolus values are calculated as $bolus = \frac{carb}{CF} + \mathbf{1}_{\{BG>TARG\}} \frac{BG-TARG}{CR}$, where $CF$, $CR$, and $TARG$ are patient-managed parameters. Due to this dependence, standard approaches to model training result in BG forecasters that are unable to disentangle the impacts of each variable and fall short of real world application when evaluated with control strategies where this correlation does not hold.

### 5.2.2 Model-based Control Solutions

BG management in T1D can be framed as a sequential decision problem. In this setting, an agent interacts with an environment (the individual as represented by CGM measurements) by taking some action (insulin bolus, potentially meal sizes) with the goal of optimizing some cumulative reward (time in the healthy BG range, or TIR). In model-based control, a model of the environment is used to test the impact of taking different actions in order to select the best action while minimizing potential harm. However, this assumes access to an accurate model. Oftentimes the data used to train the model are collected under a behavior policy in which exploration might be limited. This can lead to inaccurate model predictions when new state-action pairs are encountered during planning. Methods to address this challenge include techniques that use an explicit model of environmental uncertainty to avoid uncertain actions [Kidambi et al., 2021, Yu et al., 2020] and techniques which incorporate explicit causal or physical models of the environment to bridge the gap between the observed and unobserved state-action pairs [Arora et al., 2022, Xu et al., 2023]. While these approaches mitigate issues posed by limited exploration, they all assume that some overlap exists in the original behavior policy, i.e., that there is enough variability in action selection for a model to learn independent action effects for *in-sample* data. This assumption does not hold in many settings, such as in standard BG management, where deterministic algorithms guide actions, leading to extremely limited overlap. This makes it exceedingly difficult for models to disentangle the effects of each action on future state and reward values, hindering the learning of action counterfactuals, even within behavior policy data. Consequently, not only is there a high risk of distribution shift during deployment, but conventional methods for countering such shifts also fail because action impacts may be modeled completely inaccurately *within* the training data.

### 5.2.3 Treatment Effect Estimation Under Confounding

Learning an accurate forecaster from data with limited exploration is similar to learning treatment effects from observational data in the presence of strong confounding [Rubin, 1974, Rosenbaum and Rubin, 1983], in that we aim to recover treatment effects despite correlation between the treatment (insulin) and underlying covariates (carbohydrates) affecting the outcome (BG). Whereas randomized control trial data can simplify treatment effect estimation, randomized bolus data in BG management is unsafe, making typical treatment effect estimation methods infeasible. While techniques based on propensity score adjustment [Austin, 2011]—including matching [Imbens, 2004, Rosenbaum and Rubin, 1983], weighting [Austin and Mamdani, 2006], and deep learning-based multi-output strategies [Shalit et al., 2017, Zhang et al., 2020]—exist, they presuppose a condition of *overlap*. Overlap is present when a range of outcomes is possible for any set of covariates, but the deterministic nature of the BB strategy violates this assumption. Our method gleans insights from two non-propensity score approaches. The first is meta-learning, e.g., the S-Learner [Künzel et al., 2019], which treats treatment as a standard covariate and evaluates its effects upon its removal. The second set of approaches that we build from are techniques leveraging a small set of randomized control trial data to update models initially trained on confounded data using a residual update [Kallus et al., 2018, Hatt et al., 2022]. These latter methods learn residuals from unconfounded experimental data. Our strategy also adopts ideas from contrastive learning [Chen et al., 2020]: we utilize cosine similarity, not just for robust representations as in some methods [Kutsuna, 2023], but to ensure temporal proportionality between variable effects from different models.

Other supervised learning techniques are somewhat applicable to the challenge of modeling treatment effects under confounding, but they fall short of utility for modeling action effects within a BB behavior policy. Methods for disentangled representation learning [Wang et al., 2022], such as variational autoencoders (VAE, Kingma and Welling [2014]) and generative adversarial networks [Jeon et al., 2021], aim to separate a system's generative factors, either by encouraging independence across hidden representations or by explicitly recreating the data generating process. However, these approaches do not address strong correlation present in model inputs, resulting in an inability to isolate bolus and carbohydrate effects. Domain-knowledge approaches, which utilize known properties of the task to improve forecast accuracy, either by utilizing an explicit physiological model [Miller et al., 2020], or by enforcing directional effects [Rubin-Falcone et al., 2023], lack the specificity to disentangle bolus and carbohydrate effects. This is because multiple domain-guided choices can provide identical forecasts on fully entangled BB data. Our method does not leverage explicit domain-specific causal or physical models, nor does it rely on any randomness or exploratory data. Instead, we assume that there is a subset of actions where the influence of entanglement is slightly diminished. Our approach exploits this subset and utilizes a monotonicity assumption to disentangle the effects of paired actions.

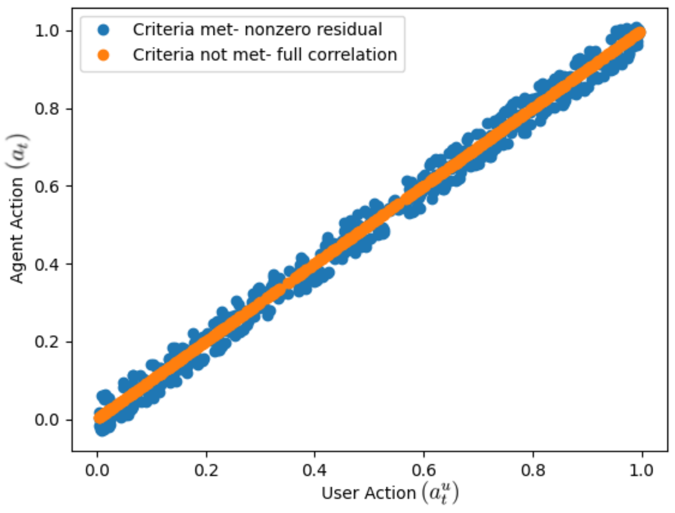
## 5.3 Problem Formalization and Notation

We model our system as a Markov decision process where, at each time point $t$, two actions are taken: the "agent action" $a_t \in \mathbb{R}$, i.e., bolus values, and the "user action" $a_t^u \in \mathbb{R}$, i.e., carbohydrate values, given a $d$-dimensional state vector $\mathbf{s}_t \in \mathbb{R}^d$. The agent controls $a_t$, while $a_t^u$ is determined by a user. Some state-dependent reward $R$ is given at each timepoint (in BG management, this reward can correspond to the inverse of glycemic risk or a similar metric). The subsequent state $\mathbf{s}_{t+1}$ is determined by some transition function T: $\mathbf{s}_{t+1} = T(\mathbf{s}_t, a_t^u, a_t)$, which a forecaster is trained to model. As is typical in observational settings, we only have access to a batch of historical data $\mathbb{D} = \{(\mathbf{s}_t, a_t^u, a_t, \mathbf{s}_{t+1}) \mid t \in \mathbb{N}\}$ collected under some behavior policy $\pi$. Within $\mathbb{D}$, actions $a_t^u$ and $a_t$ are strongly correlated, creating entanglement and complicating the task of distinguishing their individual effects on the state vector (see DAG in **Figure 5.1**).

Given $\mathbb{D}$, we aim to train a forecaster that can accurately model the independent effect of each action on the state. In particular, the approach will be evaluated via the performance of some model-based control method used to choose $a_t$ given $\mathbf{s}_t$ and $a_t^u$. If the forecaster accurately models the impact of both variables on $\mathbf{s}_t$, it can select the $a_t$ values that optimizes some reward based on $\mathbf{s}_t$.

*Behavior policy.* We assume that actions $a_t^u$ and $a_t$ are strongly linearly correlated. For all batch data $\mathbb{D}$, $a_t = Ka_t^u + f(\mathbf{s}_t)$ for some "residual" function $f$ and constant $K$. We assume $|Ka_t| \gg |f(\mathbf{s}_t)|$ generally holds, ensuring strong entanglement. We also assume that $f(\mathbf{s}_t)$ is nonzero for some actions and that $|f(\mathbf{s}_t)|$ has nonzero variation. We assume there is some criterion $C$ such that when $C$ is met, the entanglement between $a_t$ and $a_t^u$ is reduced (i.e., $|f(\mathbf{s}_t)|$ is higher for samples where $C$ is true). An example behavior policy is visualized in **Figure 5.2 Left**. In basal bolus data, the $C$ criteria is met when BG values exceed a target and a correction bolus portion is administered. The $C$ criterion can also be identified after data generation; e.g., by setting $C$ to true when $|a_t - Ka_t^u|$ is relatively large.

*Monotonicity Assumption.* We assume that the impact of action $a_t$ on the state vector $\mathbf{s}_{t+1}$ is strictly monotonic; specifically, either $\frac{d\mathbf{s}_{t+1}}{da_t} > 0$ or $\frac{d\mathbf{s}_{t+1}}{da_t} < 0$ holds for all time points and state/action values. This is a reasonable assumption in domains such as BG forecasting, where the administration of insulin boluses reliably decreases BG values in proportion to their size. This monotonic relationship ensures that the 'residual effects' of $a_t$—additional changes to the state vector resulting from increases in $a_t$ beyond a baseline—are consistent with the direction of the baseline action's impact and scale proportionally with $a_t$ value. Our approach is most effective when the second derivative $\frac{d^2\mathbf{s}_{t+1}}{da_t^2}$ is near zero: in an ideal setting where $\frac{d^2\mathbf{s}_{t+1}}{da_t^2} = 0$ for all actions, meaning that the impact of $a_t$ is linear, residual action impacts are equal to full action impacts, which is advantageous for our method. Nevertheless, our approach can accommodate a broad spectrum of second derivative values as long as monotonicity is maintained; this is empirically examined in sensitivity analyses. Example agent-action impacts ranging from ideal to expected failure are illustrated in **Figure 5.2 Bottom**.



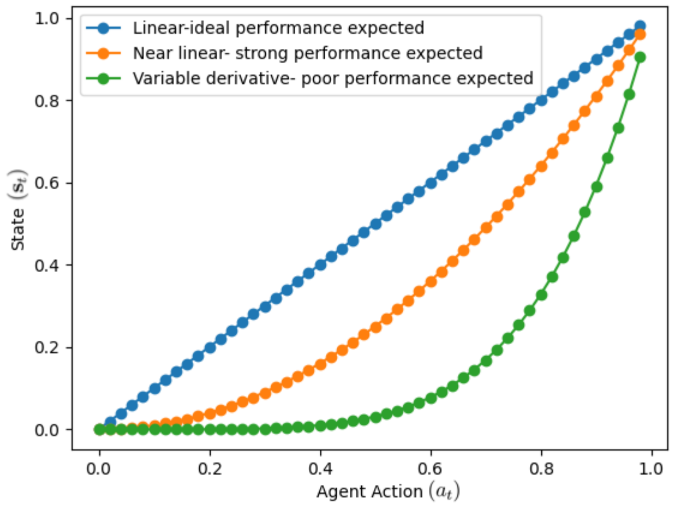


Figure 5.2: **Top:** Example data adhering to the behavior policy studied, where a strong correlation between user and agent actions is present across all samples but slightly reduced when a criterion is met. **Bottom:** Example agent action impacts on a hypothetical state variable where we expect our approach to perform optimally (blue), well (orange), and poorly (green).

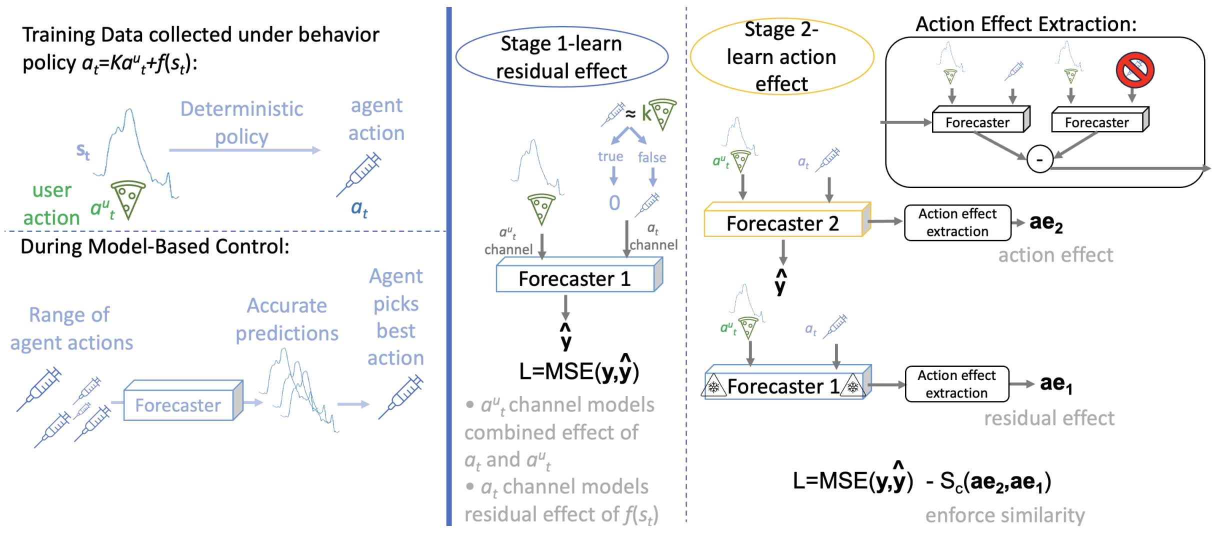


Figure 5.3: In our setting, the only available training data exhibits entanglement between paired action values. We aim to learn a forecaster on this data that can be directly applied to model-based control schemes. Our proposed approach encompasses two steps. In the first step, we separate the agent action's residual effect from the effect of the portion which is fully correlated with the fixed action. In the second step, we employ cosine similarity in the loss term to ensure that the total agent action effect learned aligns with the residual effect identified in step 1.

## 5.4 Methods

In this section, we present our proposed approach for training control-ready BG forecasters that accurately model the independent effects of carbohydrates and insulin on BG.

### 5.4.1 Proposed Approach

**Overview.** Our training strategy is organized into two stages (**Figure 5.3**). In both stages, a forecaster is trained to predict target values from a set of historical state and action data. Notably, in stage 1, we set the agent action value to zero when it is most entangled with the user action value as assessed by some predetermined criteria (e.g., if a correction dose is not administered in a BG forecasting setting). This step enables the model to isolate the impact of the residual portion of the agent action in its own channel, while the effect of the fully entangled portion of the agent action is captured in the channel modeling the user action.

In stage 2, the forecaster is retrained conventionally (no data are hidden), but we augment the loss function with cosine similarity between the effect of the agent action estimated by the current model and the (frozen) model trained in step 1. This helps encourage the model towards a representation of action effects that is proportional to and in the same direction as the residual effects without penalizing the model if the magnitudes of the residual and complete effects are different.

**Forecaster.** We assume a forecaster architecture that receives a history of state and action values (for input length T, we denote this input as $X = [\mathbf{s}_{-T:0}, \mathbf{a}^f_{-T:0}, \mathbf{a}_{-T:0}]$). The forecaster can be of any architecture, provided that each variable is input to an unique independent channel. The forecaster outputs $\hat{\mathbf{y}} = \theta(X)$, the model's forecast following the input window. We assume that the reward signal is based on a single state variable, and so our model is trained to output a time-series (for prediction horizon $h$ and reward variable $s^j$, $\hat{\mathbf{y}}$ is optimized to match $\mathbf{y} = \mathbf{s}^j_{1:h}$). While we focus on univariate time-series forecasting, our approach generalizes to multivariate prediction by simply adding additional variables to the model output and loss functions.

**Stage 1: Learning Residual Action Effects.** Our training scheme is divided into two parts. In stage 1, the forecaster is conventionally trained, with a slight modification. Whenever the condition $C$ is not met (i.e., the agent action is strongly entangled with the user action), the agent action value is set to zero. Because the agent action value is set to zero when the agent action is most correlated with the user action, the model learns the combined effect of both actions in the user action channel. The agent action channel learns the impact of the residual portion of the agent action variable, beyond what is entangled with the user action. Because the entangled portion is being modeled in a separate channel, the residual impact of the agent action variable learned at this stage should be independent of the user action and therefore easy to model. The forecast model trained in this stage is henceforth referred to as $\theta_1$, and its output is $\hat{\mathbf{y}}_1$. The loss function during this stage is the $MSE(\mathbf{y}, \hat{\mathbf{y}}_1)$.

**Stage 2: Learning Full Action Effects.** A second forecast model, $\theta_2$, is initialized from $\theta_1$. This model is trained in a standard fashion, in that no variables are removed during training and part of the loss function is the error of the model's forecast ($MSE(\mathbf{y}, \theta_2(X))$). During this stage, the residual agent action effect learned in step 1 is also utilized. Let $X_0$ denote the input data ($X$) with all agent action values set to 0. We estimate the action effect being modeled by $\theta_2$ as $\mathbf{ae}_2 = \theta_2(X) - \theta_2(X_0)$. We freeze $\theta_1$, and model the residual action effect learned by the this model as $\mathbf{ae}_1 = \theta_1(X) - \theta_1(X_0)$. While we do not expect the full agent action effect being modeled by $\theta_2$ to be similar in magnitude to the residual effect modelled by $\theta_1$, we do expect the smaller residual effect to be in the the same direction (the additional residual effect should impact the target variable positively or negatively depending on what the full action value does), and we also expect these two effects to be proportional (this follows from our monotonicity assumption). In order to utilize these assumptions, we employ cosine similarity. We do this by subtracting $S_c(\mathbf{ae}_2, \mathbf{ae}_1)$, where $S_c$ denotes cosine similarity, from the loss term. By maximizing cosine similarity, we encourage the model towards agent action effects which are proportional to and vary along with the residual effects. This reduced hypothesis space encourages better modelling of the action effects. Notably, no hyperparameters are utilized for this loss term; cosine similarity is simply subtracted directly from the MSE term.

### 5.4.2 Experimental Setup

We evaluate our proposed approach applied to the task of building control ready forecasters for BG management. In addition, we demonstrate the utility of the proposed approach in a common control task based on an inverted pendulum. We describe the implementation details of our approach and evaluation metrics in each setting.

#### 5.4.2.1 Learning Control Ready Forecasters for Blood Glucose Control

**Synthetic Data.**

We utilize a publicly available implementation of an FDA approved simulator of T1D [Man et al., 2014, Xie, 2018]. Our training dataset (BB dataset) is assembled from 20 days of data for each of the 10 adult individuals available in the simulator under a BB behavior policy. A standard meal schedule generated with the Harrison-Benedict equation [Harris and Benedict, 1919]. During forecaster training, BG values, bolus values, and carbohydrate values are used as input variables, and BG is the target to be predicted two hours into the future. Given current BG level $g$ and meal size $c$, bolus $b$ is calculated based on patient-specific parameters carbohydrate ratio ($CR$) and correction factor ($CF$): $b = \frac{c}{CF} + \mathbf{1}_{\{g>150\}}\frac{g-140}{CR}$. In this setting, $C$ corresponds to $\big(BG > Target\big)$).

*Evaluation: Accuracy of counterfactual forecasts.* While we report root mean square error (rMSE) results on held-out BB data, we do not anticipate significant performance differences across models in that setting, as our approach is designed to address out-of-sample evaluation. To evaluate model performance in a control setting, where we expect our approach to show strong performance over baselines, we developed a separate counterfactual dataset. In this dataset, 20 meals are generated for each individual, and for each meal, 20 boluses are selected randomly (from a uniform distribution across the range of observed boluse values). BG data are generated for each meal/bolus combination, essentially providing ground truth estimates of the counterfactual. Evaluation (using 2 hour prediction rMSE) is performed only in windows where a bolus is administered at the final timepoint of the input data, as these windows represent when the model would be utilized in a control setting (meal time).

*Evaluation: Impact of forecasts on control performance. Evaluation: Impact of forecasts on control performance.* In order to ground our results in a clinically actionable metric (i.e., TIR), we directly evaluate control performance on simulated data. Control performance is evaluated using a random shooting strategy: at each meal, 50 boluses sampled from the training data are selected, and a forecast is generated for each one. The bolus that yields a 2-hour forecast value closest to a target value of 140 is chosen (we also explore varying this target slightly to 150). We generate 30 days of data for each individual and each model using this control scheme and report the proportion of time that each individual (on average) spends in the target range (TIR; $70 < BG < 180$), above range (TAR; $BG > 180$) and below range (TBR; $BG < 70$).

*Evaluation: Synthetic BG Sensitivity Analyses.* We further evaluate our approach when the level of entanglement between the carbohydrate and insulin varies. We do this by modifying the behavior policy used to collect the training data. Individuals with T1D often do not adhere completely to a basal bolus strategy, but instead may occasionally administer boluses of different values or at different time-points. In this setting, instead of using a glucose-dependent correction factor, we added a random amount (sampled from a uniform distribution with a maximum value of 10) to the bolus when the blood glucose (BG) level was above the target threshold. We varied the target BG (TG) from 60 to 200 and concurrently increased the carbohydrate ratio for lower values. As a result, bolus values are calculated using the following formula: $b = \frac{c}{CF}S + \mathbf{1}_{\{g>TG\}}\mathcal{U}(0, 10(1-S))$. Here, $S = (TG-60)/140$ is a scaling factor. This scaling factor ensures that each bolus is fully random when the target value is 60 and the boluses are 100% carbohydrate-correlated when the target is 200. This allows us to modulate the correlation between $a_t^u$ and $a_t$ from nearly 0 to almost 1, providing a broad spectrum for evaluating the performance of our model compared to a standard training baseline. We expect our approach to perform similarly to baseline when the correlation is sufficiently low. To further examine the robustness of our approach to potential real BG scenarios, we systematically add unpaired bolus and carbohydrate measurements by introducing data generated with a delay between carbohydrates and boluses (up to 2 hours delayed in either direction, delay selected from a random uniform distribution). We compare our approach to baseline as unpaired boluses are added up to the point that there are more unpaired than paired samples. This experiment was selected because individuals may administer a bolus before or after meals, and having enough time between measurements could break the entanglement that our approach addresses. For both sensitivity analyses, we evaluate on the counterfactual dataset and plot average error over 10 random seeds.

*Fully Synthetic Data Sensitivity Analyses.* We generate fully synthetic data (as opposed to synthetic BG data) to test our monotonicity assumptions and the robustness of our model. A regression task is constructed with three input variables: $g$, $b$, and $c$. The objective is to predict $\hat{g}$, where $\hat{g} = g + 0.3c + f(b)$ and $f : \mathbb{R} \to \mathbb{R}$ is an agent action impact function used to explore our assumptions. Variables $g$ and $c$ are randomly sampled from a uniform distribution. We set $b = c + \mathbf{1}_{g>0.6}\mathcal{U}(0.2)$ in the training data, mimicking the basal bolus strategy. In the evaluation phase, we test the model on uniform random $b$ values, independent of $c$, to emulate our counterfactual bolus experiments. Our dataset comprises 1000 samples, split into 70%, 15%, and 15% for training, validation, and testing, respectively. The performance is evaluated against a standard forecaster and resampling (described in section 4.2.3) as baselines, and compared to training on random $b$ values to establish a lower performance bound.

Fully Synthetic Experiment 1: Examining Monotonicity Assumption. We use four $f$ functions to examine the performance of our proposed approach when the assumption of monotonicity is not met. Specifically, we test three monotonic functions: *pos*: $f(b) = 0.1b$, *neg*: $f(b) = -0.1b$, and *dis*: $f(b) = -(0.1 + 0.05\lfloor 4b \rfloor)b$. The *pos* and *neg* functions assess whether our method performs well when the user and agent action effects are in the same or opposite directions, whereas *dis* tests the method's robustness to discontinuous or non-differentiable action value functions. The final function, *sin*: $f(b) = -0.1\sin(10b)$, represents a non-monotonic agent action effect, a scenario where we do not expect our method to provide an advantage.

Fully Synthetic Experiment 2: Small Second Derivative. We utilize a series of exponential $f(b)$ functions to assess our approach's performance in relation to the second derivative of the agent action effect. By setting $f(b) = e^{\alpha b}$ and varying $\alpha$, we analyze how our method copes with different rates of change.

| | Ohio (N=12) | Tidepool (N=100) | Michigan (N=212) |
|---|---|---|---|
| Mean Age [IQR] | $\sim 47[NA]$ | 36 [14,53] | 12 [8,15] |
| % Female | 42 | 36 | 50 |
| # Individuals Included in Analysis | 8 | 65 | 8 |
| # Bolus Agreement Measurements | 17 | 184 | 3 |
| # Potential Bolus Imp. Meas. | 37 | 240 | 15 |
| TIR [IQR] | 65 [61,72] | 68 [58,84] | 44 [32,55] |
| TBR [IQR] | 4 [2,5] | 3 [1,4] | 1 [0,1] |
| TAR [IQR] | 33 [24,37] | 29 [11,40] | 55 [43.67] |

Table 5.1: Demographics information and number of measurements for each real BG dataset. Limited age information is available for the Ohio dataset. We report interquartile range (IQR) for relevant metrics, and report time in range (TIR), time below range (TBR), and time above range (TAR) for available CGM data.

**Real Data.** We employ three real-world datasets from individuals with T1D. Each dataset encompasses readings of BG, insulin, and carbohydrates from multiple subjects, spanning durations of 40-60 days. All participants were equipped with a CGM and insulin pump. To facilitate forecasting, data points were interpolated to consistent 5-minute intervals. Our selection process began with ensuring that the records had continuous 60/10/10 day sequences with a minimum of 80% completeness in CGM readings and an average of at least three daily recordings for both bolus and carbohydrate measurements. We further refined our dataset by ensuring participants had at least one bolus entry in both training and testing sets that met our specified analysis criteria. These requirements include the association of the bolus with a simultaneous carbohydrate estimate, the appropriate timing of the bolus administration/meal estimate (elaborated upon later), and the presence of continuous BG readings without any missing values for the three-hour windows following and preceding the meal. We use the following datasets; information on demographics and data availability is summarized in **Table 5.1**. The **Ohio** dataset was released for public access as a part of the Knowledge Discovery in Healthcare Data BG Level Prediction Challenge [Marling and Bunescu, 2018a]. The **Tidepool** dataset is a compilation of data from 100 participants [Neinstein et al., 2016]. Participants voluntarily opted into this dataset, leading to records that are generally comprehensive with minimal data gaps. The **Michigan** dataset draws from individuals with T1D who visited the Michigan Medicine pediatric diabetes clinic from September 2012 to December 2019. The dataset was amassed and refined for an earlier retrospective study [Rubin-Falcone et al., 2022], and this study was approved by the University of Michigan Medical School Institutional Review Board. As this data originates directly from an authentic clinical environment, it naturally exhibits more gaps compared to the other datasets.

Across these datasets we do not have access to the underlying behavior policy, as such a clear indicator of whether or not a correction bolus was given is not available. Instead, we use the following criterion to determine which actions are most entangled. For all paired boluses and carbohydrates in the training data, we fit a 1-degree polynomial function (a line) of the equation $b = xc + z$, where $b$ is bolus, $c$ is carbohydrate, and $x$ and $z$ are modelled for each individual. We consider the actions to be less entangled when the bolus value is sufficiently far from that line, i.e., $C = \big(|b - xc - z| > \chi\big)$, where $\chi$ is a tuned hyperparameter. While there is a strong relationship between bolus and carbohydrate values in real data, the basal bolus strategy is not strictly followed. This means that unpaired boluses and carbohydrates (i.e., boluses that occur without a carbohydrate estimate, and vice-versa) exist in the dataset. We utilize only paired boluses and carbohydrates for step 1 of our proposed approach and then include the full dataset in step 2. The full dataset is used for the baseline forecaster. We also only include "well-timed" boluses in step 1, as defined below, which ensures a cleaner sample. Finally, to ensure that the directionality of the learned bolus effect is correct, during step 1, we replace non-zeroed bolus values with their distance from the best-fit line $(b - xc - z)$.

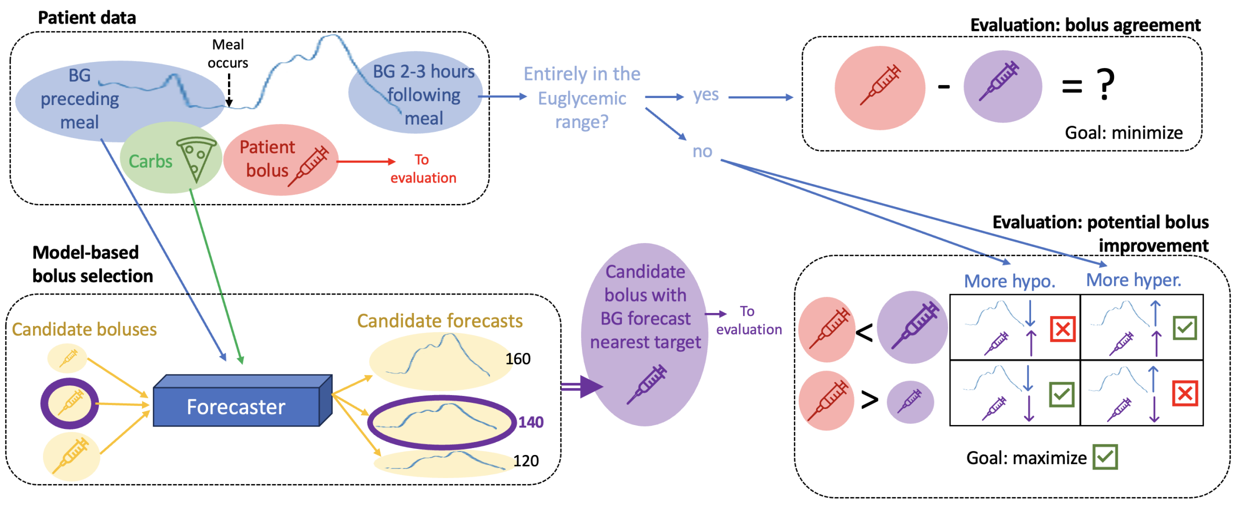
*Evaluation: proxy metrics.* Directly assessing a forecaster's ability accurately forecast BG under counterfactual actions is not feasible with only observational data, and directly applying the control methods studied here prospectively is too risky. Instead, we introduce two metrics to determine the forecaster's efficacy for BG management within a rudimentary control framework (**Figure 5.4**). At evaluation, for every bolus paired with a meal in the test dataset, an alternative bolus is selected through the forecaster. Similar to evaluation on simulated data, 100 counterfactual values are randomly generated from a uniform distribution ranging from zero to the largest bolus in the observed training data. The bolus that aligns its 2-hour forecast closest to a target of 140 is selected. This chosen bolus is then compared with the one selected by the patient.

Metrics include:

Bolus Agreement. This metric evaluates how the forecaster would perform for meals where patients successfully managed their BG. This metric is therefore only calculated for meals where the BG remains entirely within the euglycemic range during the 2 to 3 hours following the meal. The metric is the mean absolute error between the patient-chosen bolus ($p$) and the model-chosen bolus ($m$), scaled by the individual's average bolus value ($\mu$): $|p - m|/\mu$. A lower score suggests the forecaster is in line with a successful patient's decision, while a higher score indicates a potential disparity.

Potential Bolus Improvement. Taking inspiration from the Clarke error grid [Clarke et al., 1987], this metric evaluates the forecaster's potential to outperform the patient. As such, it is only calculated for windows where BG in the 2 to 3 hour period following the meal leaves the euglycemic range for at least one time point. BG trajectories are categorized by their outcome (hyperglycemic or hypoglycemic), and boluses are divided based on whether the model's choice is greater or lesser than the patient's. The goal is to determine the proportion of bolus adjustments that are in the 'well-calibrated' direction. A model is considered well-calibrated if it suggests larger boluses than the patient when the predominant outcome is hyperglycemic and lower boluses when the outcome is hypoglycemic. The metric's value ranges from 0 (indicating poor performance) to 1 (indicating ideal performance), with 0.5 suggesting random choices. While this metric does not measure optimal real-world control performance, it can help identify systems that are likely to perform more poorly than human judgment.

Assessments are exclusively made on "well-timed boluses": instances where a bolus and carbohydrate are recorded concurrently, and where BG remains relatively stable before a



Patient data
Meal occurs
BG preceding meal
BG 2-3 hours following meal
Carbs
Patient bolus
To evaluation
Entirely in the Euglycemic range?
yes
no
Evaluation: bolus agreement
- = ?
Goal: minimize
Evaluation: potential bolus improvement
More hypo.
More hyper.
<
>
Goal: maximize
Model-based bolus selection
Candidate boluses
Forecaster
Candidate forecasts
160
140
120
Candidate bolus with BG forecast nearest target
To evaluation

Figure 5.4: Evaluation methodology for real data: We input historical BG and carbohydrate data into our forecaster alongside an array of potential boluses. The bolus that yields a forecast closest to the desired target is chosen. If the post-meal BG remains within the euglycemic range, we compute bolus agreement by measuring the difference between the patient's bolus choice and our model's recommendation. Conversely, if the BG veers outside of the euglycemic range we determine potential bolus improvement by assessing the fraction of model-recommended boluses that lean in the optimal direction—higher than the patient bolus when post-meal BG is hyperglycemic and lower if hypoglycemic.

meal (less than a 20 point increase in the three hours preceding the meal) and sees at least a 40-point increase in the three hours following the meal. This ensures evaluations are not biased by misaligned bolus timings. We do not evaluate any bolus measurement where another bolus or meal appears in the 90 minutes following the evaluated bolus to circumvent confounding.

We validate each metric using simulated data, assessing the correlation between each metric and counterfactual forecast error. This analysis reveals the extent to which these metrics are indicative of potential control capability.

#### 5.4.2.2 Learning Control Ready Forecasters for Inverted Pendulum Control

We assess our method's viability in other model-based control contexts using a standard reinforcement learning (RL) benchmark, the inverted pendulum [Arora et al., 2022, Lambert et al., 2021, Lee et al., 2020, Wang et al., 2019]. This setting serves to demonstrate our approach's applicability to physics based systems where the second derivative of the agent action impact on the state might have higher variability. This setting differs from BG management in that full and residual action impacts may be both positive and negative. Implemented via Mujoco using default settings [Todorov et al., 2012], the task involves moving a cart to balance a pole vertically on top of it, with system variables being the pole's angle ($\phi$) and angular velocity ($\omega$), and the cart's speed and position ($v$ and $x$). The dataset $\mathbb{D}$ is created from a policy based on an angle-responsive force with a correction proportional to the angular velocity when a threshold is exceeded, inspired by the basal bolus strategy and a simple cart-pole solution [Xu, 2021]. The force $F$ is defined as $F = 12\phi + \{0.15\omega$ if $|\omega| > 1.5\}$, with values optimized for performance. This policy is able to keep the pole elevated for over 200 time steps. Given its strong correlation with $\phi$ (r=0.99), distinguishing the impact of the push on the cart ($F$) from gravity's $\phi$-proportional influence becomes challenging (here, gravity's force is a hidden user action which is also correlated with $\phi$). The goal here is learning a control-ready forecaster from $\mathbb{D}$, produced through this constrained control scheme, without added exploration. VAE and up-weighting are used as baselines, as domain-knowledge methods are inapplicable. Each model predicts $\phi$ from past values of $\phi$,$F$,$\omega$,$x$, and $v$.

*Implementation Details.* Given the behavior policy studied here, strongly entangled samples are present when the magnitude of the angular velocity of the pole is below a set threshold (i.e., C=$\big(|\omega| > 1.5\big)$). To ensure that the directionality of the learned residual agent action effect is correct, during step 1, we replace non-zeroed agent action values with their distance from the would-be fully entangled action value (i.e., we set the agent action to $F - 12\phi$). A single timestep is used as a prediction horizon in this setting. As such, the cosine similarity loss term in step 2 of our proposed approach is computed across batch elements, rather than across timesteps.

*Evaluation.* Models are assessed on reserved behavior policy data and on control for 100 random runs, noting mean forecast error during control and timesteps to failure. Control involves sampling 200 action values from a uniform distribution centered at 0 and choosing the force associated with the lowest magnitude pole angle at the next step. Here counterfactual actions are examined when selected by the model.

#### 5.4.2.3 Performance Target and Baselines

*Performance Target- Counterfactual Training.* A counterfactual *training* dataset is developed. This dataset is generated in a similar fashion to the BB dataset used for training other models, but the bolus at each meal is selected randomly (as in the counterfactual dataset used for evaluation). Our primary baseline model was trained on this data to establish a performance target. We assume that a model which accurately disentangles bolus and carbohydrate effects would perform similarly to a model trained with this ideal, though unrealistic, data.

*Baselines.* We compare our method against seven baselines including domain knowledge utilization, treatment effect estimation, and disentangled representation learning approaches (**Table 5.2**). Although these approaches offer some relevance to our context, we hypothesize that their limitations in handling the lack of overlap in BB data reduce their effectiveness towards disentangling the effects of carbohydrates and bolus on BG.

| Baseline | Domain Knowledge | Treatment Effects | Disentangled Representations |
|---|---|---|---|
| Standard | | | |
| LED | $\chi$ | | |
| VAE | | | $\chi$ |
| SUP | $\chi$ | | |
| VAE+SUP | $\chi$ | | $\chi$ |
| Resample | | $\chi$ | |
| Matching | | $\chi$ | |

Table 5.2: Categorizations of baselines.

**1. Standard Training:** This method involves the training of the base model without any additional supervision or modifications.

**2. Linked Encoder/Decoder (LED):** This model was developed to incorporate domain knowledge regarding the direction of bolus and carbohydrate effects into a recurrent forecasting approach [Rubin-Falcone et al., 2023]. This approach works by utilizing individual decoders for each variable and gating inputs so that only relevant samples are passed to the appropriate decoders. Bolus and carbohydrate values are passed to their relevant decoders, as well as to the encoder, via a skip connection. Given the simultaneous occurrence of boluses and carbohydrates in our dataset, the gating mechanism is less effective. To address this, we only input boluses and carbohydrates into their respective decoders (and not the encoder), which enforces better separation of effects. This approach ensures that the effects of boluses and carbohydrates can only be learned by their relevant decoders, which, due to the incorporation of domain knowledge, should have opposite effects on the output.

**3. Variational Auto-Encoder (VAE):** In this approach, we use a probabilistic encoder and aim to minimize the Kullback-Leibler (KL) divergence between the distribution of each element in the hidden state and the unit normal. Previous research has shown that this method can lead to disentangled hidden state representations [Kingma and Welling, 2014]. We include this approach to investigate whether a disentangled hidden state leads directly to a disentangled representation of entangled input variables.

**4. Domain Supervision (SUP):** This method involves the addition of an auxiliary loss term during training. While training the forecaster, we estimate the effects of boluses and carbohydrates as the difference in forecasts when each variable is hidden from the model (i.e., where $X_B$ and $X_C$ denote the input data with boluses and carbohydrates set to zero, bolus effects are $\theta(X) - \theta(X_B)$ and carbohydrate effects are $\theta(X) - \theta(X_C)$). We add the loss term $\alpha \times$ (bolus effect $-$ carbohydrate effect) to encourage the model to output a negative bolus effect and a positive carbohydrate effect, which is in line with our domain knowledge. The value of $\alpha$ is tuned using counterfactual evaluation data in order to assess the best possible performance this model can achieve.

**5. VAE + SUP:** This approach combines the techniques of VAE and supervision. We include this model to test the hypothesis that a disentangled hidden state, when combined with additional supervision towards correct bolus and carbohydrate effects, can lead to accurately modeled effects.

**6. Matching:** This approach matches meals that include a correction portion to the bolus (BG>TG) with meals of similar size (measured in grams of carbohydrates) that do not include a correction portion. We exclude meals that cannot be paired.

**7. Resample:**In this approach, each input sample that includes a correction bolus is included in the batch N times. Similar to SUP, the value of N was determined by tuning on counterfactual evaluation data in order to achieve the strongest possible performance.

#### 5.4.2.4 Implementation Details

**Architecture Details.** Each network is implemented as a single layer bidirectional LSTM with 25 hidden units. For BG data, we use an input window of 36 timesteps, equivalent to 3 hours, which represents the approximate maximum duration of bolus/carbohydrate effects. A 24-timestep window, or 2 hours, is selected as our prediction horizon since it provides a long enough period for control without posing excessive challenge. For the inverted pendulum, we utilize an input of six timesteps. This is because 36 timesteps would retain excessive, potentially irrelevant information. Considering the immediate effects of each action and the continuous control nature of the task, where an action is taken at every timestep—unlike our BG management scenario—a single timestep is deemed more appropriate for the prediction horizon.

**Training.** We implement and train our models in Pytorch 1.9.1 with CUDA version 10.2, using Ubuntu 16.04.7, a GeForce RTX 2080, an Adam optimizer [Kingma and Ba, 2014], a batch size of 500 and a learning rate of 0.01. We train for at least 100 iterations, and then until validation performance does not improve for 50 iterations, selecting the model for which validation performance was best. Because a larger amount of data was utilized for the real dataset, we use a patience 10 iterations and a minumum of 20 iterations. We decrease the learning rate to 0.001 during stage 2 of training for our proposed approach.

This value was selected from [0.01,0.001,0.0001,0.00001] and tuned to behavior policy (BB) validation performance using adult#001. For the sensitivity analysis with unpaired boluses and carbohydrates, a model is trained on the paired data using our proposed approach and then fine-tuned using the unpaired data, which allows our approach to better utilize the unpaired boluses and carbohydrates. For real data experiments, the value of $\chi$ (the parameter which determines which boluses are set to zero during stage 1 of training) was individually tuned for each participant. The tuning aimed to minimize [bolus agreement - potential bolus improvement] on the training and validation data. The values considered were 0.01, 0.1, and 0.2 times the mean bolus value of each individual's training and validation data.

**Data**. A unique model is used for training and testing for each individual. We split each dataset into training, validation and test sets used for evaluation purposes. For the inverted pendulum, we use 3 runs, 1 each for training, validation and testing. For simulated BG data, for each individual, we use 14 days for training and 3 each for validation and testing. For the Ohio dataset, we split the training data into 80% training and 20% validation, and use the held-out test data for evaluation. For the other two real datasets, we use 60 days for training, and 10 each for validation and testing. To maximize the number of paired bolus and carbohydrate values, we group proximate meals and boluses. Carbohydrate estimates within 15 minutes of a bolus are shifted to coincide with the bolus time. Similarly, boluses within 15 minutes of another are combined at the earlier time point. The same method is applied for carbohydrate values. These are performed as preprocessing steps before all analyses. For both the baseline and proposed approaches, during training and evaluation, we ignore windows with missing BG values in either the input data or prediction horizon.

## 5.5 Results

Throughout our experiments, we aim to answer the following questions:

- Does training a BG forecaster on BB data using our approach yield improvements when forecasting under counterfactual actions? (Section 5.5.1- Forecasting Results)
- Do these counterfactual forecasts lead to better decisions and in turn better control, when compared to models trained with relevant baselines? (Section 5.5.1- Control Results)
- How does our approach perform when boluses and carbohydrates are not perfectly paired or correlated? (Section 5.5.1- Sensitivity Analyses)
- Does our approach offer potential benefit for real data BG control? (Section 5.5.2)
- Does our approace lead to impvoed inverted pendulum control? (Section 5.5.3)

| | In-sample Error | Out-of-sample Error |
|---|---|---|
| Model | BB rMSE (↓) | Counterfactual Action rMSE (↓) |
| Standard | 31.4 [29.3,33.6] | 49.2 [46.5,51.9] |
| LED | 33.1 [30.9,35.3] | 43.6 [41.1,46.1] |
| VAE | 35.6 [33.5,37.8] | 48.7 [45.9,51.4] |
| SUP | 32.0 [29.8,34.1] | 46.4 [44.0,48.9] |
| VAE+SUP | 35.2 [32.9,37.5] | 47.1 [44.5,49.7] |
| Resample | 32.7 [30.4,34.9] | 46.9 [44.4,49.5] |
| Matching | 32.8 [30.7,35.0] | 46.1 [43.5,48.8] |
| Proposed | 32.0 [29.9,34.2] | 34.0 [31.9,36.0] |
| Counterfactual Training | 33.8 [31.7,35.8] | 23.5 [21.8,25.1] |

Table 5.3: Simulated BG data forecasting results. Mean forecasting error (rMSE) on basal bolus (BB) and counterfactual data are shown. Outcomes are reported as: value [95% confidence interval (CI)]. Our proposed approach outperforms baselines for the counterfactual dataset. CIs were calculated using 1,000 bootstrap samples. Abbreviated baselines include the linked encoder-decoder (LED), a variational autoencoder (VAE), and magnitude-based domain supervision (SUP).

### 5.5.1 Results on Synthetic Blood Glucose Data

We evaluate control performance on synthetic BG management using a counterfactual dataset and by assessing model-based control directly. We also examine the sensitivity of our approach as our assumptions vary.

| Model | %TIR (↑) | %TAR (↓) | %TBR (↓) |
|---|---|---|---|
| Standard | 44.2 [43.4,45.0] | 54.0 [53.3,54.7] | 1.8 [1.6,2.0] |
| LED | 31.6 [30.8,32.3] | 68.0 [67.3,68.7] | 0.5 [0.4,0.5] |
| VAE | 53.6 [52.7,54.6] | 45.3 [44.4,46.3] | 1.0 [0.9,1.2] |
| SUP | 44.3 [43.4,45.1] | 54.1 [53.3,54.9] | 1.6 [1.5,1.8] |
| VAE+SUP | 50.7 [49.7,51.7] | 48.2 [47.3,49.2] | 1.1 [1.0,1.2] |
| Resample | 37.0 [36.2,37.9] | 61.9 [61.1,62.8] | 1.1 [0.9,1.2] |
| Matching | 43.8 [42.9,44.7] | 53.8 [52.9,54.7] | 2.4 [2.2,2.6] |
| Proposed | 81.1 [80.3,81.9] | 13.9 [13.2,14.6] | 5.0 [4.6,5.4] |
| Proposed (Target=150) | 72.0 [71.0,72.9] | 25.6 [24.8,26.5] | 2.4 [2.2,2.7] |
| Counterfactual Training | 81.5 [80.7,82.3] | 12.9 [12.3,13.5] | 5.6 [5.2,6.1] |
| BB Control | 79.7 [78.9,80.4] | 17.8 [17.1,18.5] | 2.5 [2.3,2.7] |
| BB Control +20% error | 72.6 [71.7,73.4] | 26.6 [25.7,27.4] | 0.9 [0.8,0.1] |
| BB Control +40% error | 64.7 [63.8,65.7] | 35.0 [34.1,35.9] | 0.3 [0.2,0.3] |

Table 5.4: Simulated BG control results. Time in, above, and below range (TIR, TAR, TBR), when using the relevant approach in a random shooting control scheme are shown. Outcomes are reported as: value [95% confidence interval]. Our proposed approach outperforms baselines for TIR. CIs were calculated using 1,000 bootstrap samples.

**Forecasting Results (Synthetic BG Data).** Our method outperforms baseline techniques, showing reduced forecast error on the counterfactual dataset (**Table 5.3**): while all models perform similarly on data collected under a BB behavior policy, ours strongly outperforms the next-best baseline on counterfactual data (rMSE 34.0 vs. 43.6). Treatment estimation methods (Resample, Match) and supervision techniques (LED, SUP) perform slightly better than the VAE forecasting method but fall short of our approach.

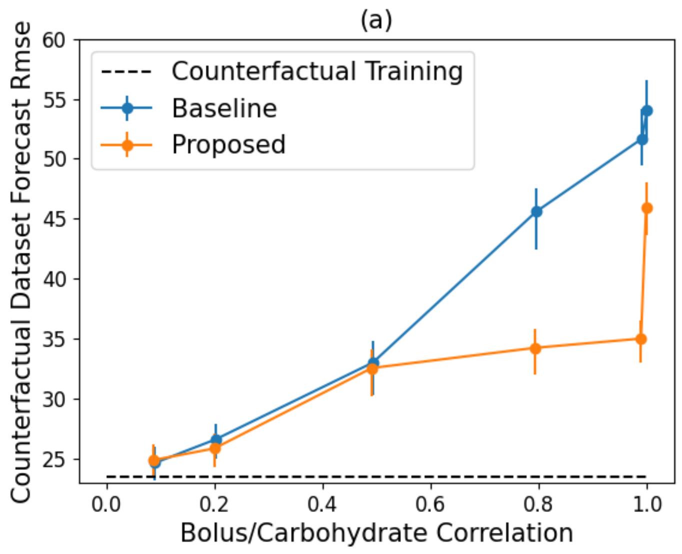
**Control Results (Synthetic BG Data).** With respect to BG control, the proposed forecasting approach achieves 81.1% time in range when paired with a simple control algorithm, far exceeding the 44.2% time in rage achieved by the standard forecaster and the 53.6% time in range achieved by VAE, the next best baseline (**Table 5.4**). Our approach paired with the simple control algorithm utilized here slightly improves over a standard BB controller (81.1% TIR vs 79.7%). The BB strategy requires an individual to accurately estimate a carbohydrate ratio and correction factor. When these are not set appropriately (e.g., 20% or even 40% error) the time in range achieved by the basal bolus control strategy drops to 72.6% at 20% error and 64.7% at 40% error. Our approach did not offer improvement in reducing hypoglycemia, only achieving a performance of 5% TBR. However, this severe hypoglycemia is abated when a target of 150 is used for bolus selection. In this setting, the proposed approach achieves a TBR comparable to BB control, although TIR is reduced. A two-hour prediction horizon may not fully encompass the duration of bolus effects. Therefore, setting a higher target can compensate for additional blood glucose (BG) decreases linked to the chosen bolus, which are not accounted for in the model.

**Synthetic BG Sensitivity Analyses.** As we alter the degree of correlation between boluses and carbohydrates in the behavior policy data, our approach generally maintains stable performance, while baseline performance steadily degrades for more entangled data (See **Figure 5.5**). As expected, at low r-values ($< 0.5$) the baseline model and our approach perform similarly. As we increase the proportion of paired boluses and carbohydrates in the training data, our proposed approach begins to significantly outperform the baseline forecasting approach when more than half of the sample is unpaired.

**Fully Synthetic Data Sensitivity Analyses.** In our simple synthetic data regression task, as predicted, our method significantly outperforms the baselines for the three monotonic agent action impact functions (*pos*, *neg*, and *dis*), nearly reaching the lower performance bound for the first two functions, which due to their linear nature, are ideally suited for our approach (**Figure 5.6 top**). For the non-monotonic *sin* function, our method does not show improvement over the baseline. As demonstrated in **Figure 5.6 bottom**, and in line with our expectations, our approach performs on par with or surpasses the lower performance bound when $\frac{d^2\hat{g}}{db^2}$ is small. However, as $\frac{d^2\hat{g}}{db^2}$ increases, performance begins to deteriorate, eventually falling below that of the baseline for significantly large values of $\frac{d^2\hat{g}}{db^2}$.

### 5.5.2 Results on Real Blood Glucose Data

We utilize proxy metrics to asses the potential control ability of our approach compared to a standard baseline forecaster using real data. We validate these metrics on simulated data before demonstrating our approach's benefit on real world data.



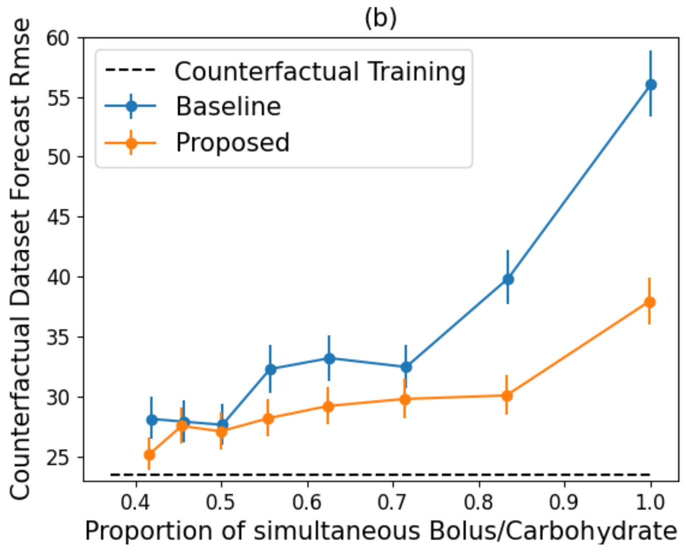
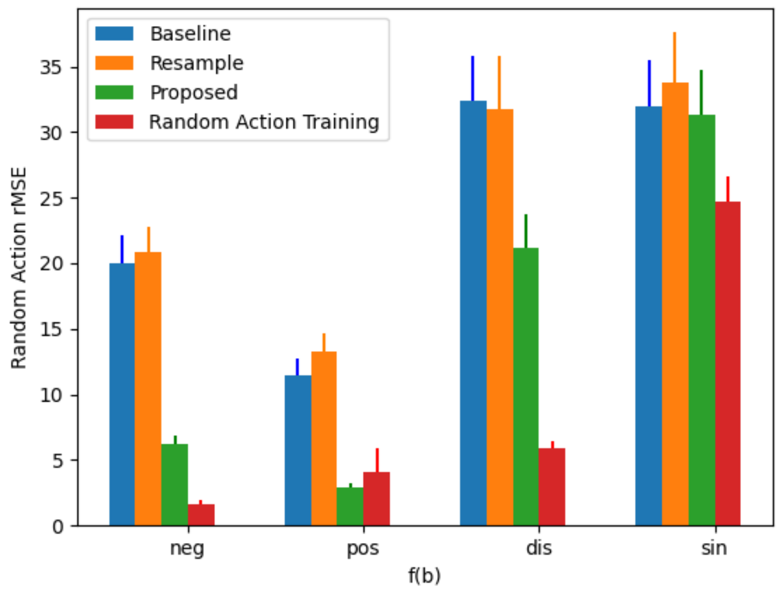


Figure 5.5: Sensitivity analyses on synthetic BG data. (a) Our approach vs. baseline as the correlation between bolus and carbohydrate values varies. Our method markedly outperforms the baseline when the correlation between bolus and carbohydrate values exceeds r=0.5 and yields similar performance to the baseline at low correlation values. (b) Our approach consistently outperforms baseline when unpaired bolus and carbohydrate values are added to the training data. Evaluation for both plots utilize the counterfactual dataset.



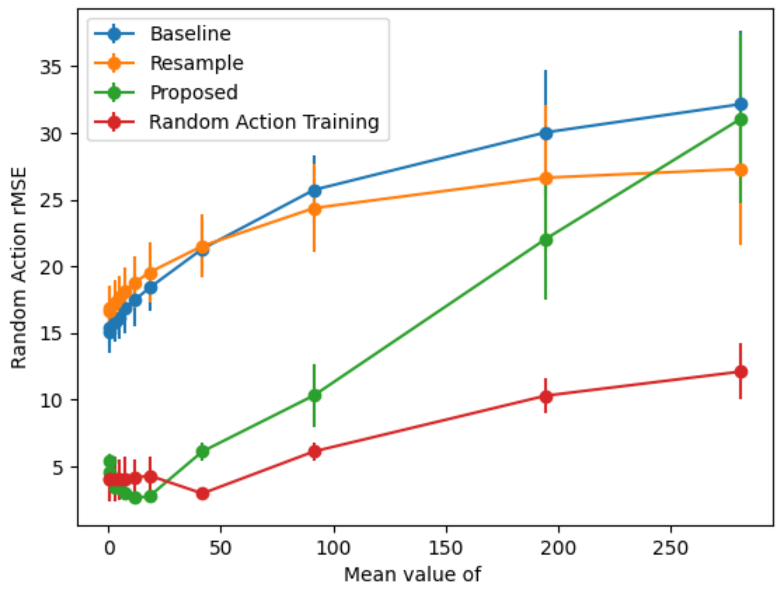


Figure 5.6: Model performance for fully synthetic data in a simple regression task. Top: the proposed approach significantly outperforms baselines when applied to monotonic agent action effects but does not yield improvement in scenarios where monotonicity is violated. Bottom: Model performance as a function of the average magnitude of the second derivative of the agent action effect, as assessed by a family of exponential functions. The proposed approach markedly outperforms the baselines when the second derivative is of low magnitude, indicating a near-linear relationship, but performance deteriorates as the magnitude increases.

**Validation of Evaluation Metrics.** We validated the proposed evaluation metrics: bolus agreement and potential bolus improvement, using simulated data. When trained on simulated BB data, our proposed approach demonstrates lower bolus agreement and higher potential bolus improvement than baseline, and the outcomes are further improved when the forecaster is trained on counterfactual data (see **Table 5.5**). We examined the Pearson correlation between counterfactual bolus rMSE and potential bolus improvement across the 10 individuals and 3 models (baseline-BB, proposed-BB, and counterfactual bolus), and found that higher error was significantly associated with lower potential bolus improvement (r=-0.47, p=0.01). We also found that counterfactual bolus rMSE and bolus agreement were significantly correlated (r=0.44, p=0.02). This indicates that better results with these metrics likely translate to superior control performance.

**Potential Control Performance Results (Real BG Data).** Our three datasets consisting of data from real patients are summarized in **Table 5.1**. While our approach does not notably improve overall forecast performance compared to the baseline on behavior policy data, it does show promise in both proxy metrics (**Table 5.5**). Notably, there is a pronounced improvement in potential bolus improvement across all three datasets (baseline potential bolus improvements are 0.73, 0.54, and 0.53, compared to the proposed approach: 0.86, 0.60, and 0.80). Additionally, bolus agreement sees a marked improvement for the Tidepool dataset, the largest and most robust collection. For the Ohio dataset, the improvement in bolus agreement is modest. On the the Michigan dataset, although bolus agreement is higher for our approach, this metric was calculated with a limited sample of 3 measurements where BG was entirely within the euglycemic range 2-3 hours post-meal. In addition to containing fewer boluses for evaluation, the Michigan dataset presents weaker forecast performance, probably because this clinic-based dataset contains BG data that is much noisier and more poorly controlled compared to other datasets (exhibiting an average TIR approximately 20% lower than the Tidepool or Ohio datasets), making it a more complex task to understand BG dynamics in this setting. Finally, we note that in all real datasets, the performance, as measured by both proxy metrics, falls between the results of models trained on simulated BB data and those trained on simulated counterfactual data, indicating some entanglement, although less than observed in the fully entangled simulated BB data, as expected.

| Model | Forecast rMSE (↓) | Potential Bolus Improvement (↑) | Bolus Agreement (↓) |
|---|---|---|---|
| SIMULATED | | | |
| BB Training, Baseline | 31.4 [29.3,33.6] | 0.48 [0.31,0.69] | 0.98 [0.80,1.17] |
| BB Training, Proposed | 32.0 [29.9,34.2] | 0.93 [0.83,1.00] | 0.55 [0.42,0.70] |
| counterfactual Training | 33.8 [31.7,35.8] | 0.97 [0.90,1.00] | 0.41 [0.31,0.51] |
| REAL | | | |
| Ohio- Baseline | 50.9 [47.8,54.3] | 0.73 [0.57,0.86] | 0.82 [0.61,1.03] |
| Ohio- Proposed | 50.4 [47.8,53.6] | 0.86 [0.76,0.95] | 0.81 [0.62,1.00] |
| Tidepool- Baseline | 51.7 [48.4,55.2] | 0.54 [0.47,0.60] | 0.89 [0.78,1.00] |
| Tidepool- Proposed | 52.8 [49.4,56.1] | 0.60 [0.54,0.66] | 0.82 [0.74,0.91] |
| Michigan- Baseline | 72.1 [64.0,81.8] | 0.53 [0.27,0.80] | 0.61 [0.29,0.84] |
| Michigan- Proposed | 72.3 [63.0,83.5] | 0.80 [0.60,1.00] | 0.74 [0.64,0.88] |

Table 5.5: Results for real data experiments. Mean forecasting error (rMSE) on behavior policy data, mean bolus agreement, and mean potential bolus improvement are shown. Outcomes are reported as: value [95% confidence interval]. Both metrics are validated in simulated data, and our proposed approach outperforms baselines for most relevant metrics.

### 5.5.3 Inverted Pendulum

During control, our proposed forecaster achieves significantly lower error compared to all baselines (rMSE=0.70 vs. best performing baseline of 2.59, **Table 5.6**). As a result, only the proposed forecast model is able to achieve good control when used in model-based control (353 timesteps until failure versus 11). To examine why other forecasters fail, we plotted the forecast error during control as a function of the absolute difference between the selected action and the action that would have been selected under the behavior policy (**Figure 5.7**). Our approach exhibits stable forecast performance. However, the baseline standard forecaster's error drastically increases when the action selected is far from the behavior policy action. The forecasting error of the baseline is nearly perfectly correlated with selected action distance from control policy action, indicating that this model has not learned a representation of the effect of the action independent of the angle of the pole.

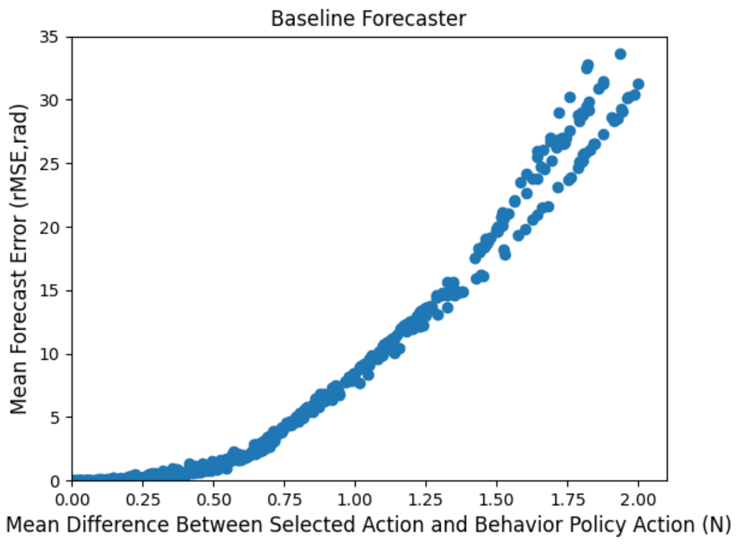
| Model | Behavior Policy rMSE (↓) | During-control rMSE (↓) | Mean Num. Steps Until Failure (↑) |
|---|---|---|---|
| Behavior Policy | N/A | N/A | 252.14 (248.65,255.53) |
| Standard forecaster | 0.35 (0.28,0.42) | 2.81 (2.68,2.94) | 11.64 (11.48, 11.80) |
| Upweight | 0.37 (0.29,0.44) | 2.83 (2.69,2.96) | 11.62 (11.43, 11.86) |
| VAE | 0.33 (0.27,0.39) | 2.59 (2.48,2.69) | 11.70 (11.54, 11.90) |
| Proposed | 0.74 (0.73,0.74) | 0.71 (0.70,0.72) | 353.82 (337.29,370.08) |

Table 5.6: Results for inverted pendulum experiments. Mean forecasting error (rMSE) on behavior policy data, during random-shooting model-based control, and mean number of epochs before failure when using each approach for control are reported. rMSE reported in degrees. Outcomes are reported as: value [95% confidence interval]. Our proposed approach outperforms baselines.

## 5.6 Discussion & Conclusion

Our proposed forecasting approach accurately models the independent effects of bolus insulin and carbohydrates on BG despite only training on data in which the two actions are strongly correlated. It outperforms several baselines including treatment-effect estimation techniques, disentangled representation learning approaches, and domain knowledge utilization, with respect to out-of-sample forecasting error. In addition, the better forecasting performance translates to better time in range when paired with a simple control strategy. Taken together with our inverted pendulum results, our BB control findings indicate that our approach has the potential to improve over the behavior policy with *zero additional exploration or data*, provided a suitable control policy is employed. Through sensitivity analyses we demonstrate that our approach remains beneficial over baselines as our assumptions vary. Additional simulated BG sensitivity analyses and real data results with proxy metrics indicate that our approach could be beneficial in a real BG setting.

Our approach offers a potential alternative to BB control. ML-based forecaster-driven model-based control holds several advantages over BB management: while BB control requires manual parameter adjustments by patients, a ML-based controller can auto-update with physiological changes, reducing patient involvement. Furthermore, if a patient mises-



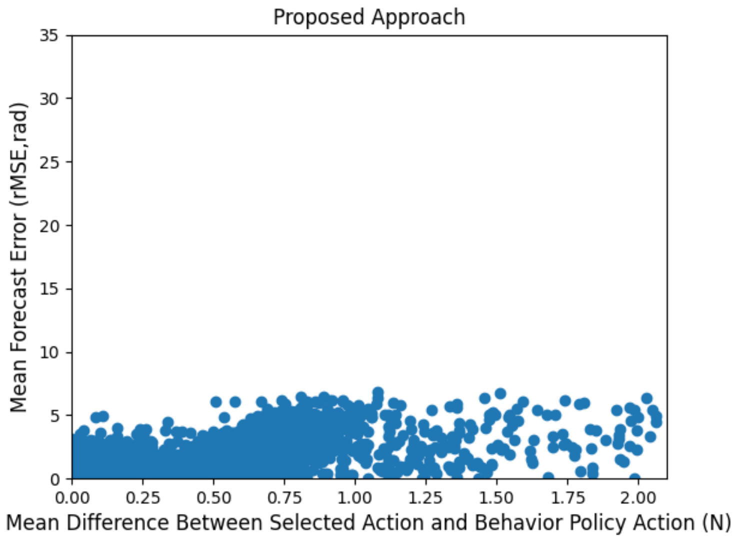


Figure 5.7: Inverted pendulum forecast error during model-based control as a function of mean difference between the selected action and the would-be behavior policy action for baseline (top) and the proposed approach (bottom). The baseline forecaster fails when it selects actions dissimilar to what was observed in the training data.

timates control parameters, or fails to update them as their body's glucoregulatory system changes, their control performance will suffer. A forecaster updated with online learning would avoid this error as patient-parameter selection would be obviated.

Current automated BG management strategies include rule-based forecaster control strategies [Ahmed et al., 2020] and model-free RL [Fox et al., 2020]. Rule-based methods are not self-updating, and model-free RL demands extensive exploratory data, which is impractical. Our method is noteworthy as model-based control systems are more sample efficient compared to model-free RL [Atui, 2015], and machine learning offers more adaptability than rule-based methods [Wang et al., 2020]. Our method is the first to initialize an adaptable forecaster using BB BG data, representing a significant advance in harnessing the efficiency and adaptability of ML-based forecasters for BG management

Evaluating counterfactuals using real data is always challenging. To this end, we proposed and validated proxy metrics to gauge a BG forecaster's potential in a control setting. Our proposed forecasting approach demonstrates improvement over the baseline in these metrics (where we had sufficient statistical power for evaluation). However, these metrics merely hint at possible control performance. The full effectiveness of our algorithm in real-world BG control cannot be ascertained without a real-life trial, which comes with inherent risks. Although our results suggest that our method can surpass baselines, implementing a deep forecaster for BG management needs further refinement. Even with the improvement offered by our approach, the observed bolus agreement and potential bolus improvement values are not near optimal (0 for bolus agreement and 1 for potential bolus improvement). As such, the proposed approach is not yet ready to be considered for deployment. Several avenues could enhance our model. First, the BG target value used to select boluses during evaluation (140) was picked to match the simulator and to provide a reasonable contrast between models, but in practice, patient-specific targets could be more appropriate and lead to improved performance in a real-world scenario. Other potential improvements include extending forecast horizons to three or four hours, adopting sophisticated planning methods beyond random shooting, and integrating more training data or implementing online learning. These enhancements could bring us closer to algorithms capable of safely managing real BG levels. We note that while our forecaster's effectiveness with a random shooting strategy is demonstrated, further assessment is needed with advanced control methods like model-based RL. Another limitation is our dependency on "well-timed" boluses during training, leading to the exclusion of many individuals. Effectively training a ML-based BG forecaster requires access to a well-curated collection of past data for each individual. We trained our models using 50 days of data. Access to a comparable dataset, complete with CGM measurements and accurately recorded meal sizes, would be necessary for model deployment in clinical settings. Our approach is built on a relatively lightweight architecture, so access to a smartphone or similar computing resources would be sufficient to train an individual-specific model. A final limitation of note is that, in simulated control experiments, our proposed approach has a higher TBR than the BB approach (5.0% vs 2.5%), marking a potential danger that will need to be addressed before an approach like ours can be utilized. We note that a longer prediction horizon, though more challenging to train for, could abate this, because two hours is too short to fully capture bolus impacts. Additionally, when a BG target of 150 is used for bolus selection, TBR with our proposed approach is comparable with BB control, indicating that careful individual-tailored target selection has the potential to balance TIR and TBR in future studies.

In model-based control, forecasters may fail to learn accurate effects if actions are highly correlated in the behavior policy. This could hamper a forecaster's applicability to action selection during control. We propose an approach that addresses this challenge. Our approach enables the training of control-ready forecasters utilizing only data generated under a behavior policy with strongly correlated action pairs. Our method functions when a linear correlation between paired actions is strong but imperfect and when the impact of one action is strictly monotonic. Beyond the BG control setting, our technique could prove useful in other settings with highly correlated actions. For instance, our approach could demonstrate utility for predicting energy demand from multiple sources like residential, commercial, and industrial sectors, where usage patterns often overlap and influence each other, or forecasting action impacts from adjacent parts in a robotics setting.

CHAPTER 6

# Conclusion

This dissertation addressed open challenges regarding the utilization of data-driven techniques for blood glucose (BG) management in type 1 diabetes (T1D). Although automated approaches for this task have been proposed, they require frequent management of model parameters [Collyns et al., 2021, Ware et al., 2022], and accurate meal-size estimates [Brazeau et al., 2012, Mehta et al., 2009], both of which are burdensome for individuals. We propose techniques for mitigating these burdens: automated meal-size denoising could ease the required accuracy of patient estimates, and improved reliability for machine-learning based forecasters has the potential to allow this automatically adjustable solution to be utilized for BG management. We presented approaches that address these challenges in BG management which also have applications beyond this field. We summarize challenges and our contributions.

First, we noted the challenge of counting carbohydrates in BG management, which is required for patients to select appropriate boluses. We formalized this problem and identified it as a general challenge present with wearables: utilizing an auxiliary data stream to denoise noisy, univariate patient estimates. In order to ease the burden, we proposed a technique that utilizes denoising autoencoders [Vincent et al., 2008] and co-teaching [Han et al., 2018, Yu et al., 2019] to iteratively filter and denoise carbohydrate estimates by using CGM datastreams. As demonstrated in **Chapter 3**, our approach showed promise for denoising carbohydrate estimates in real patient data, and has potential applications to problems like remote pain and mood monitoring.

Second, we identified a challenge present when forecasting BG values: the **sparse but important variable problem.** In **Chapter 4**, we formalized the problem, which occurs when important auxiliary variables are significantly sparser than the target variable being predicted. This makes incorporating these variables into forecasters challenging. Our proposed approach, the linked encoder-decoder, improved utilization of boluses and carbohydrates in BG forecasters. It is also potentially applicable to domains like traffic, stock price, and vital sign forecasting.

Finally, we identified a second challenge preventing the utilization of machine learning-based forecasting approaches for blood glucose management. The deterministic basal bolus control scheme creates a strong correlation between carbohydrate and bolus values, which makes learning their impacts challenging. In **Chapter 5**, we proposed an approach which accounts for this challenge and utilizes correction boluses to more accurately model the impacts of these variables. We demonstrated our approach's benefit for controlling simulated BG, and proposed proxy metrics to demonstrate its application to real BG data. We also evaluated our approach in the inverted pendulum setting, and it has potential application in other treatment effect estimation and robotics settings.

There are several areas in this dissertation that could be interesting for further examination. First, while our carbohydrate denoising approach has the capability of recovering clean estimates, which can be utilized to improve time in range (TIR) longitudinally, it requires 90 minutes of BG values following the meal as currently presented. It may be possible to recover clean meal size estimates with less data, provided a technique were highly sensitive to BG variation. Such a technique would allow for real time correction doses to be calculated. Subsequently, incorporating our approach into an automated control method could improve performance by cleaning the carbohydrate signal for more reliable bolus selection. We also have not investigated the applicability of our approach to recovering unrecorded carbohydrate measurements. Our approach is easily adaptable to this setting, provided some regularization is used to avoid over imputation. Our approach has the potential to obviate carbohydrate counts altogether, provided a sensitive enough algorithm can be developed, but much work remains on this front.

Second, while our approach for addressing the SIV problem has been validated in an RNN-based approach, it has not been utilized in other architectures. Since we first completed work on the SIV problem, RNNs have lost their state-of-the-art status to transformers. Auxiliary-variable-sparsity problems, similar to the SIV problem, have been identified in transformers, with solutions including binning [Labach et al., 2023] and triplet feature representation approaches [Tipirneni and Reddy, 2021]. Our proposed approach is suitable for any encoder-decoder based model, and so could be applicable to many transformer setups, potentially combined with an existing orthogonal approach. Examining the efficacy of our approach paired with a time-series transformer presents an interesting direction, specifically towards the application of real BG data. While we have identified several domains where the SIV problem applies (including vital signs, traffic, and stock price forecasting), we have not yet evaluated our approach beyond BG, and as such this presents an interesting direction for further study.

Finally, while we have proposed an approach for learning control-ready forecasters from basal bolus data and demonstrated its benefit when applied to real data, our approach is still far from being potentially applicable to a real individual. Our approach outperforms baseline on both bolus agreement and potential bolus improvement, indicating that it has better potential to manage BG. However, our proposed approach is still far from ideal in terms of these metrics. More work on longer term prediction, more accurate forecasts, and better model-based control algorithms is likely necessary before any deep forecaster will be precise enough in terms of bolus effect modelling to be truly useful for direct BG management.

The main contributions of this dissertation are: 1) developing a novel technique for recovering noisy patient estimates in the presence of an auxiliary data stream and 2) proposing novel techniques for addressing auxiliary variable sparsity and confounding in a forecasting setting. Going forward, we expect the techniques developed in this dissertation to help build robust and automatically adjustable machine learning models for BG management.

# APPENDIX A

# Appendix for Denoising Autoencoders for Learning from Noisy Patient-Reported Data

## A.1 Simulated Dataset Details

During data generation, days where a patient either had more than 25 timepoints of glucose at the minimum value of 40, or more than 35 timepoints over 450 were thrown out for being non-realistic. The meal schedule used to generate simulated data was based on the Harrison-Benedict equation [Harris and Benedict, 1919] as implemented in [Fox et al., 2020]. In our simulation, for all datasets generated, we used the default basal-bolus controller from the existing implementation of the simulator to administer insulin, but we delayed five sixths (randomly selected) of the bolus administrations up to 3.5 hours, with delay time randomly sampled from a uniform distribution. The delay allows for disentanglement between carbohydrate and bolus effects. 20% of carbohydrates are not reported, to make the dataset more realistic, as missingness is common.

## A.2 Tuning Details

For each model, tuning was performed on simulated adult#001 using validation performance. No additional tuning was performed for other individuals or noise functions. For Noisier2Noise, we selected $\alpha$, the parameter that controls the relative noise distributions, from [0.1,0.3,0.5,0.7,0.9,1.0,1.25,1.5,1.75,2], ultimately selecting $\alpha = 1$. Because we do not assume access to the exact noise we would not expect this method to perform spectacularly, but note that it often outperforms other baselines.

For co-teaching methods, we performed a simple grid search over the values of $E_k$=[250,500] (where 500 is the minimum number of training iterations), $\tau$=[0.333,0.5,0.667], and $\sigma$ = [0.1,0.3,0.5,0.7]. For $\mathbf{N^{+}2N}$, we selected $E_k = 250$, $\tau = 0.333$, and $\sigma = 0.1$. For the supervised setting co-teaching (**SUPCT**), we set selected $T_k = 500$, $\tau = 0.5$ and $\sigma = 0.3$.

Hyperparameter options were selected from a limited but comprehensive spectrum of values that cover a reasonable search space (given that all hyperparameters are limited to a fixed interval) without consideration for task. Only a small number of options were considered to avoid computational burden, as a simple grid search was used. A ground truth signal was used for evaluation during tuning, which is a limitation, as such a signal is generally not available in real-world scenarios. However, we note that we did not re-tune for each individual (tuning to simulated adult#001), nor did we retune for the real-world dataset, which is substantially different from the simulated dataset. Proxy measures such as CRC may also be used for tuning.

## A.3 Additional Training Details

We split each dataset into training, validation and test sets used for evaluation purposes. For the simulated dataset, we use 80 days for training, 20 for validation, and 50 for testing. For the real dataset, we split the training data into 80% train and 20% validation. The held-out test data were used for evaluation only. We implement and train our models in Pytorch 1.9.1 with CUDA version 10.2, using Ubuntu 16.04.7, a GeForce RTX 2080, an Adam optimizer [Kingma and Ba, 2014], and a batch size of 500. We use a learning rate of 0.01 and a weight decay of $10^{-7}$. We train for at least 500 iterations, and then until validation performance does not improve for 50 iterations, selecting the model for which validation performance was best.

For both datasets, we train and test a model on each individual and report across-individual averages. Such individual-specific models/evaluations are common in blood glucose control and forecasting [Silvia Oviedo, 2016], since dynamics vary greatly across individuals and individual-specific training data are typically available.

We perform co-teaching on samples containing non-zero $y$ values only. However, when training all models (including baselines) we also pass zero-valued $y$ samples (and their corresponding $\mathbf{b}$ values) through both DAEs and take loss equal to $\hat{y}^2$ for these samples. We do this because there are many more samples with zero-valued carbohydrates than there are with positive values, and this allows the models to learn from this larger collection. We report results on only positive-valued $y$ values, because denoising is only applied to such values.

## A.4 Alternate Noise Functions

We consider noise functions that might arise in carbohydrate counting. None are highly dissimilar from our main analysis noise function: we aim here at feasibility, rather than a comprehensive survey on a broad selection of loss functions, which our method would likely be unable to address without further tuning or modification. Here, $\mathcal{U}(a, b)$ denotes a uniform distribution with values between $a$ and $\mathbf{b}$. Carbohydrate values range between 0 and 200. After adding noise, $y$ values are capped above and below by 1 and 200. Alternate noise functions include:

1. Zero-mean multiplicative Gaussian: $y = (1 + \mathcal{N}(0, .75))x$

2. Negative-mean multiplicative Gaussian (primary noise function): $y = (1 + \mathcal{N}(-.25, .5))x$

3. Zero-mean additive Gaussian: $y = x + \mathcal{N}(0, 40)$

4. Negative-mean additive Gaussian: $y = x + \mathcal{N}(-30, 50)$

5. Zero-mean multiplicative Uniform: $y = \mathcal{U}(.5, 1.5)x$
6. Negative-mean multiplicative Uniform: $y = \mathcal{U}(0, 1.6)x$
7. Zero-mean additive Uniform: $y = x + \mathcal{U}(-60, 60)$
8. Negative-mean additive Uniform: $y = x + \mathcal{U}(-60, 40)$

## A.5 Sensitivity to noise in the $b$ signal.

Although the auxiliary **b** signal is expected to be relatively low noise compared to $y$, some noise is possible. In our motivating domain, CGM data contain a non-negligible amount of noise. In the simulator, this noise is modeled as additive time-varying Gaussian [Man et al., 2014]. To evaluate our approach's sensitivity to noise in the auxiliary signal, we added an increasing multiplier to the noise term in the CGM for each simulated individual. The multiplier ranged from 1X to 6X, with 1X being the standard CGM. At 6X noise the magnitude of the signal is more than 25% noise on average, and the original glucose signal is barely detectable (**Figure A.1**).

We found that at each noise setting, our approach outperformed all baselines (**Figure A.2**). Also encouragingly, even with 20% noise, our approach performs similarly to the best baseline trained on clean data. Taken together, this indicates that our approach is robust to noise in the relatively clean auxiliary signal, up to levels more than four times what is typically observed.

## A.6 Sensitivity to hyperparameter $\tau$

In order to examine the impact of including various amounts of data in our final training sample, we varied hyperparameter $\tau$, which controls the proportion of samples within each minibatch that the networks are backpropagated on. We note that low values of $\tau$, corresponding to a larger, noisier, sample, result in relatively stable performance, with all values

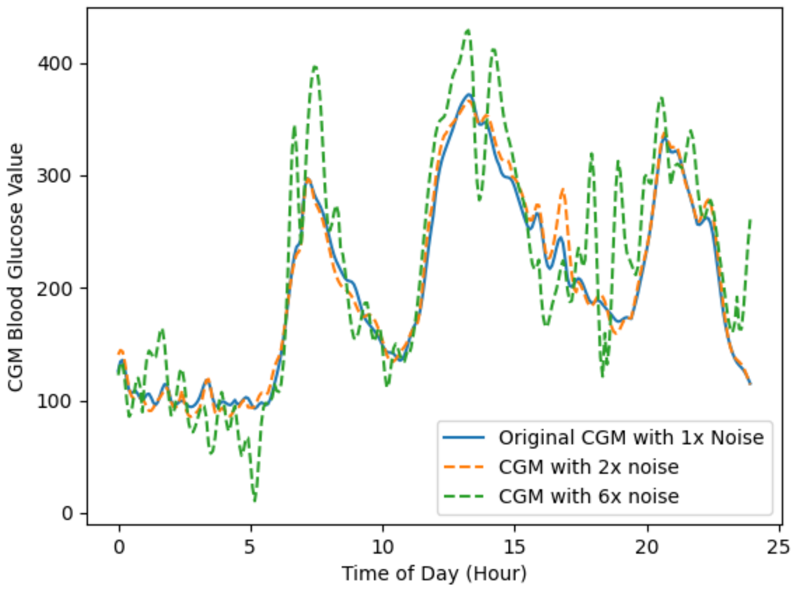


Figure A.1: One day's worth of blood glucose data for simulated subject adult#002 with 1 x and 5 x additional simulator noise added. At 5 x, the signal is almost unrecognizable.

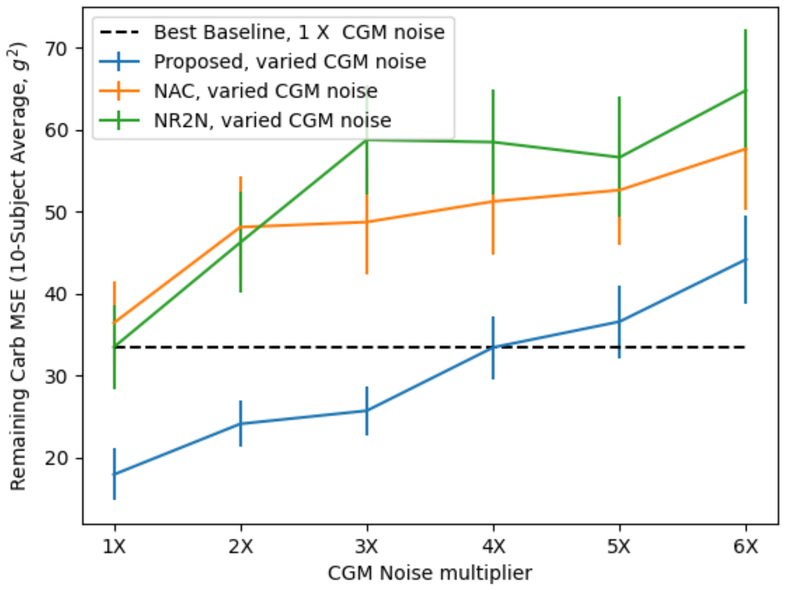


Figure A.2: Our approach vs. strongest baselines for varying levels of noise in the CGM signal, average across all 10 simulated individuals with 1,000 sample bootstrap SEs as error bars. Our approach performs well even when noise is fairly large.

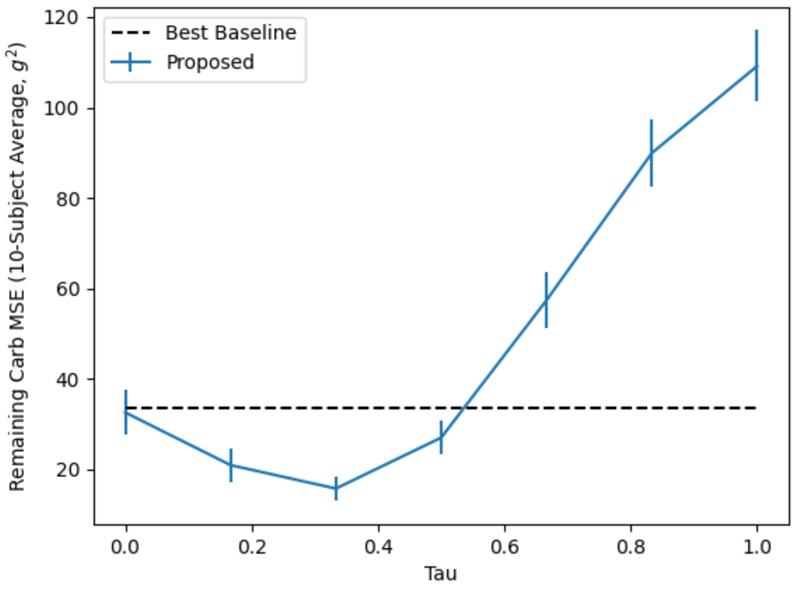


Figure A.3: Model performance as a function of hyperparameter $\tau$, with all else constant, across all 10 simulated individuals. Our model outperforms baseline as long as fewer than 50% of samples are excluded.

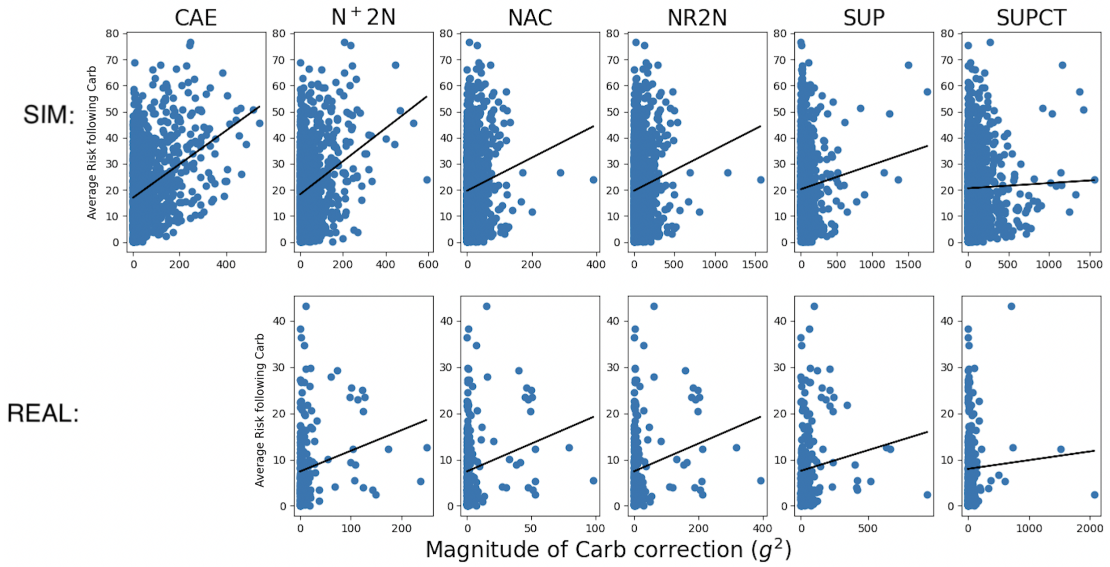


Figure A.4: Risk following the carbohydrate vs. magnitude of carbohydrate correction learned for all models and both datasets. Besides the clean autoencoder, **N$^+$2N** performs best.

between zero and 0.5 (half of the samples excluded) offering substantial improvement over the best performing baseline (**Figure A.3**). Performance degrades for larger values of $\tau$, which likely indicates that the approach is robust until the final training sample becomes too small to be effective.

## A.7 CRC Plots

With $\mathbf{N^{+}2N}$, we see a higher correlation between the magnitude of carbohydrate correction and risk following the meal compared to baselines for both real and simulated data (**Figure A.4**).

# APPENDIX B

# Appendix for Forecasting with Sparse but Informative Variables

## B.1 Ohio Dataset Experiments

### B.1.1 Data and Training Description

This dataset includes both the OHIOT1DM 2018 and 2020 datasets, developed for the Knowledge Discovery in Healthcare Data Blood Glucose Level Predication Challenge Marling and Bunescu [2020a]. The data pertain to 12 individuals, each with approximately 10,000 5-minute samples for training and 2,500 for testing, with carbohydrate administrations occurring every 88 timepoints on average, (median, [IQR]: 70, [56,134]), and insulin boluses occurring every 52 timepoints on average (36, [28,63]). 12% of glucose values are missing, but we do not include windows with missing glucose values.

This dataset contains the same variables and is processed and analyzed identically to the simulated dataset, except as described here. For the real dataset we evaluated on the held-out test data from the challenges. The remaining data were split into 80% train and 20% validation. Models were trained for at least 25 epochs, and then until validation data performance did not improve for 10 epochs.

The Ohio Dataset (Ohio T1D Blood Glucose Level Prediction Challenge, 2018 and 2020), can be made available through a data-use agreement with the owners:

http://smarthealth.cs.ohio.edu/OhioT1DM-dataset.html.

### B.1.2 Primary Results

On the Ohio dataset, performance gains are more moderate (rMSE 20.16 vs 20.36, **Table B.1**), when compared to the simulated dataset. Multiple approaches exhibit negative SIV usage for the Ohio dataset, indicating that including the SIVs does more harm then good. We hypothesize that this is due to noise in the carbohydrate signal.

| Model | rMSE [95%CI](Usage) | MAE [95%CI](Usage) |
|---|---|---|
| Enc/Dec | 20.36[19.4,21.3](0.08) | 14.67[14.1,15.2](0.24) |
| SIV Fine-tune | 21.74[20.9,22.6](-1.30) | 16.25[15.7,16.9(-1.35) |
| SIV Initialize | 20.98[20.0,22.0](-0.54) | 14.99[14.4,15.6](-0.09) |
| Full Capacity | 20.98[20.0,21.9](-0.54) | 15.09[14.5,15.7](-0.18) |
| Proposed | 20.16[19.3,21.1](0.28) | 14.64[14.1,15.2](0.27) |

Table B.1: Forecasting Error and SIV usage for the real dataset. Outcomes are reported as: Error [95% confidence interval] (SIV Usage). Our proposed approach outperforms baseline, although to a lesser degree than the simulated dataset. Confidence intervals were calculated from bootstraps with 1,000 resamples.

### B.1.3 Individual Level Results and Ablations

With respect to the Ohio data, while the overall trend was the same as the simulated data, across individuals, the correlation between baseline error and our approach's improvement over baseline was not significant (r=0.22, p=0.49, **Figure B.1 (b)**). We hypothesize that this again might be due to the presence of noise in the carbohydrate signal, which prohibits our model from accurately modeling the SIV signal (explored in Section 5.5). Alternatively, the intrinsic dynamics in the Ohio dataset may simply be more complex and thus result in more variability across individuals. The association between baseline encoder/decoder SIV usage and improvement over baseline does hold for real data (**Figure B.1** (a)), Pearson r=-0.59, p=0.042,

The restriction element is important for the Ohio dataset (**Table B.2**, rMSE increases to

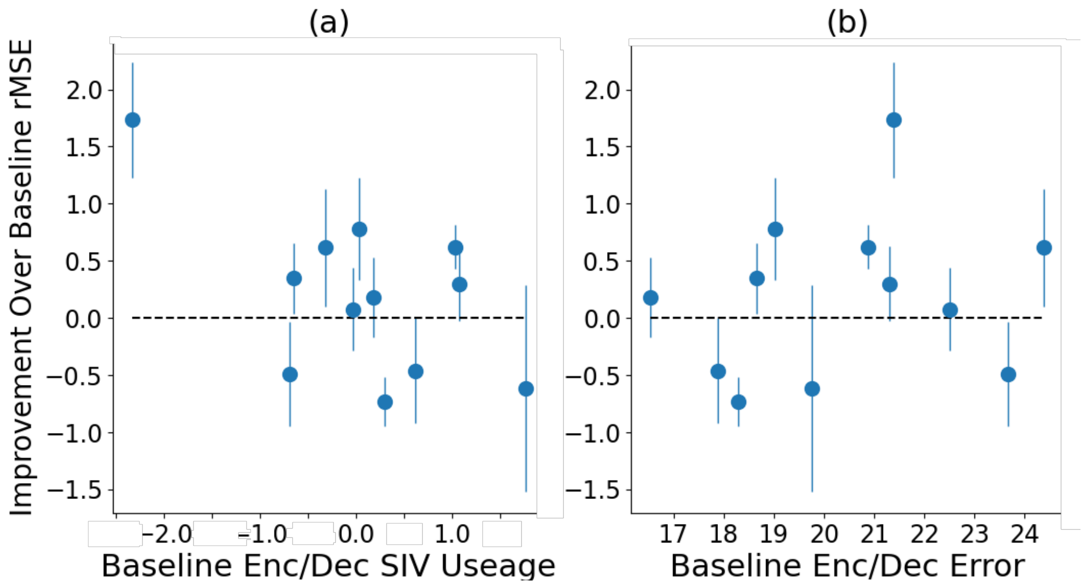


Figure B.1: (a) Our architecture's improvement over the encoder/decoder baseline vs baseline SIV usage for the Ohio dataset. Our method's benefit increases as baseline SIV usage decreases. (b) Improvement over baseline vs baseline prediction error for Ohio data, for each individual. Improvement over baseline is not correlated with baseline error.

20.38 from 20.16 when restriction is removed). This is likely because this dataset presents a more difficult challenge, compared to the simulated dataset, due to noise in the SIV signal and more complex target variable dynamics. For the Ohio dataset, we see a decrease in performance for each ablation. Our architecture works by isolating the effect that the SIV signal has on the target variable and enforcing consistency with domain knowledge. Although the domain knowledge is very general (we only restrict the signal direction), it improves performance, offering a benefit over isolation alone for the Ohio dataset. More restrictive model guidelines, such as directly restricting the architecture to use a detailed physiological model, could be beneficial, but during model development, we found that "less is more," in that a small amount of restriction with significant flexibility was most effective. However, some sort of domain-knowledge-based-guidance is helpful to overcome the challenges posed by the SIV problem, since without it, it is difficult to learn anything useful from the small number of non-zero samples.

| Model | rMSE [95%CI](Usage) | MAE [95%CI](Usage) |
|---|---|---|
| | Ohio | |
| No Gating | 20.34[19.5,21.3](0.11) | 14.77[14.2,15.3](0.13) |
| No Restriction | 20.38[19.5,21.3](0.06) | 14.73[14.2,15.3](0.18) |
| No SIV Input | 20.71[19.8,21.6](-0.27) | 14.97[14.4,15.6](-0.07) |
| Only SIV Input | 20.52[19.6,21.5](-0.08) | 14.70[14.1,15.3](0.20) |
| Proposed | 20.16[19.3,21.1](0.28) | 14.64[14.1,15.2](0.27) |

Table B.2: rMSE and MAE, with SIV usage, for each ablation. Outcomes are reported as: Error [95% confidence interval] (SIV Usage). Confidence intervals were calculated from bootstraps with 1,000 resamples.

## B.2 Impact of Carry-Forward Approach

Utilizing the Carry-forward approach improves performance on both datasets for both the baseline encoder/decoder and our proposed approach (**Table B.3**).

| Model | rMSE [95%CI](Usage) | MAE [95%CI](Usage) |
|---|---|---|
| | Simulated- Carry Forward | |
| Enc/Dec | 15.63[14.1,16.9](11.13) | 12.42[11.1,13.6](6.63) |
| **Proposed** | **13.07[11.8,14.2](13.69)** | **10.45[9.4,11.4](8.61)** |
| | Simulated- NO Carry Forward | |
| Enc/Dec | 16.46[14.6,17.8](10.30) | 12.97[11.5,14.1](6.09) |
| Proposed | 16.08[14.5,17.4](10.68) | 12.80[11.4,13.9](6.25) |
| | Ohio- Carry Forward | |
| Enc/Dec | 20.36[19.5,21.3](0.08) | 14.67[14.1,15.2](0.24) |
| **Proposed** | **20.16[19.3,21.1](0.28)** | **14.64[14.1,15.2](0.27)** |
| | Ohio- NO Carry Forward | |
| Enc/Dec | 20.64[19.7,21.5](-0.20) | 14.98[14.4,15.6](-0.08) |
| Proposed | 20.41[19.5,21.4](0.03) | 14.85[14.3,15.4](0.05) |

Table B.3: Forecasting Error and SIV usage for both datasets, examining our primary baseline and proposed approach with and without our carry-forward approach. Outcomes are reported as: Error [95% confidence interval] (SIV Usage). Both methods benefit from utilizing the carry-forward approach on both datasets. Confidence intervals were calculated from bootstraps with 1,000 resamples.

| Region | Baseline | Proposed |
|---|---|---|
| A | 97.1 | 97.8 |
| B | 2.36 | 1.43 |
| C | 0.00 | 0.00 |
| D | 0.05 | 0.08 |
| E | 0.00 | 0.00 |

Table B.4: Proportion of points in each region of the Clarke Error Grid. Region A represents strong forecasts, while regions C through E represent potentially catastrophic errors

## B.3 Clarke Error Grid

A Clarke error grid demonstrates where a forecaster could lead to catastrophic failure; predicted BG values are compared to true values, and regions where making treatment decisions based on forecasts would lead to poor health outcomes are highlighted. Clarke error grids for our approach and the best performing baseline on the simulated dataset, are shown in **Figure B.2** and **Figure B.3**, respectively. Both approaches demonstrate fairly strong performance, but our approach has 98% of points in region A, while the baseline has 97% of points in region A (**Table B.4**). Region A represents the region where utilizing the forecasts for BG control would reliably lead to good health outcomes. While one percentage point is a modest improvement, every prediction is important in a clinical setting. This illustrates that while our approach is not a complete solution to reliable BG forecasting, it a step in the right direction.

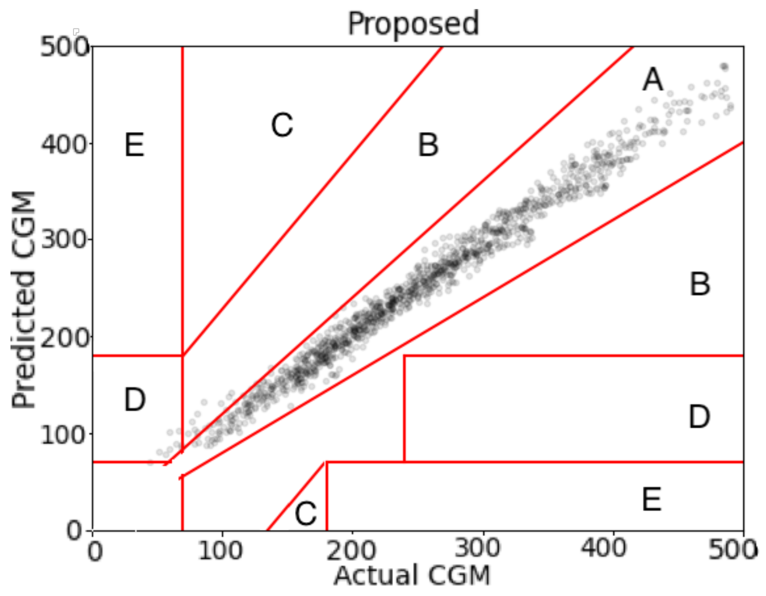


Figure B.2: Clarke error grid for the proposed approach.

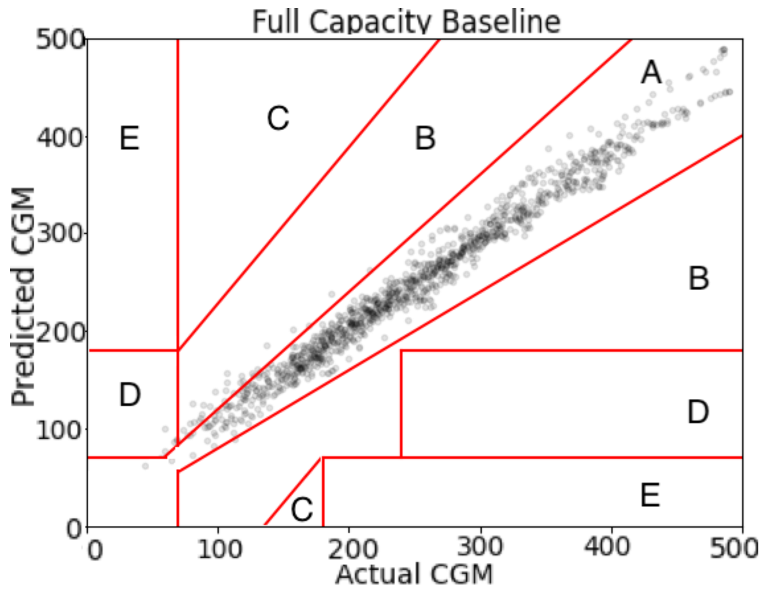


Figure B.3: Clarke error grid for the full capacity baseline.

# BIBLIOGRAPHY

Syed Haris Ahmed, David L. Ewins, Jane Bridges, Alison Timmis, Nicola Payne, Cormac Mooney, and Claire MacGregor. Do-it-yourself (diy) artificial pancreas systems for type 1 diabetes: Perspectives of two adult users, parent of a user and healthcare professionals. *Adv Ther.*, 2020.

Oren Anava, Elad Hazan, Shie Mannor, and Ohad Shamir. Online learning for time series prediction, 2013.

Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning, 2021.

Mohammadreza Armandpour, Brian Kidd, Yu Du, and Jianhua Z. Huang. Deep personalized glucose level forecasting using attention-based recurrent neural networks. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2021. doi: 10.1109/IJCNN52387.2021.9533897.

Rushiv Arora, Bruno Castro da Silva, and Eliot Moss. Model-based reinforcement learning with sindy, 2022.

Kavosh Asadi Atui. *Strengths, Weaknesses, and Combinations of Model-based and Model-free Reinforcement Learning*. PhD thesis, 2015.

Peter C Austin. An introduction to propensity score methods for reducing the effects of confounding in observational studies. *Multivariate behavioral research*, 46(3):399–424, 2011.

Peter C Austin and Muhammad M Mamdani. A comparison of propensity score methods: a case-study estimating the effectiveness of post-ami statin use. *Statistics in medicine*, 25 (12):2084–2106, 2006.

Joshua Batson and Loic Royer. Noise2self: Blind denoising by self-supervision. *ICML*, 2019.

Kirstine J. Bell, Carmel E. Smart, Garry M. Steil, Jennie C. Brand-Miller, Bruce King, and Howard A. Wolpert. Impact of fat, protein, and glycemic index on postprandial glucose control in type 1 diabetes: Implications for intensive diabetes management in the continuous glucose monitoring era. *TYPE 1 DIABETES AT A CROSSROADS*, 2015.

Anne-Sophie Brazeau, H Mircescu, Katherine Desjardins, C Leroux, I Strychar, J.M. Ekoé, and R Rabasa-Lhoret. Carbohydrate counting accuracy and blood glucose variability in adults with type 1 diabetes. *Diabetes research and clinical practice*, 99, 2012.

Claudia Bull, Joshua Byrnes, Ruvini Hettiarachchi, and Martin Downes. A systematic review of the validity and reliability of patient-reported experience measures. *Health Services Research*, 2019.

Defu Cao, Yujing Wang1, Juanyong Duan, Ce Zhang, Xia Zhu, Conguri Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Spectral temporal graph neural network for multivariate time-series forecasting. *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Kanad Chakraborty, Kishan Mehortra, Chilukuri Mohan, and Sanjay Ranka. Forecasting the behavior of multivariate time series using neural networks. *Neural Networks*, 5:961–970, 1992.

Yi Chang, Luxin Yan, Meiya Chen, Houzhang Fang, and Sheng Zhong. Two-stage convolutional neural network for medical noise removal via image decomposition. *IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT*, 2020.

Chris Chatfield. In *Time-Series Forecasting*. Chapman and Hall/CRC, 2000.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020.

Kate Churruca, Chiara Pomare, Louise A. Ellis, Janet C. Long, Suzanna B. Henderson, Lisa E. D. Murphy, Christopher J. Leahy, and Jeffrey Braithwaite PhD. Patient-reported outcome measures (proms): A review of generic and condition-specific measures and a discussion of trends and issues. *Health Expectations*, 2021.

WL Clarke, D Cox, LA Gonder-Frederick, W Carter, and SL Pohl. Evaluating clinical accuracy of systems for self-monitoring of blood glucose. *Diabetes Care*, 1987.

OJ Collyns, RA Meier, ZL Betts, DSH Chan, C Frampton, CM Frewen, NM Hewapathirana, SD Jones, A Roy, B Grosman, N Kurtz, J Shin, RA Vigersky, BJ Wheeler, and MI de Bock MI. Improved glycemic outcomes with medtronic minimed advanced hybrid closed-loop delivery: Results from a randomized crossover trial comparing automated insulin delivery with predictive low glucose suspend in people with type 1 diabetes. *Diabetes Care*, 2021.

Emanuele D'Amico, Rocco Haase, and Tjalf Ziemssen. Review: Patient-reported outcomes in multiple sclerosis care. *Multiple Sclerosis and Related Disorders*, 2019.

Syama Sundar Rangapuram Emmanuel de Bezenca an, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella Hilaf Hasson, Patrick Gallinari, and Tim Januschowski. Normalizing kalman filters for multivariate time series analysis. *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Ian Fox, Lynn Ang, Mamta Jaiswal, Rodica Pop-Busui, and Jenna Wiens. Deep multi-output forecasting. *KDD*, 2018.

Ian Fox, Joyce Lee, Rodia Pop-Busui, and Jenna Wiens. Deep reinforcement learning for closed-loop blood glucose control. *Proceedings of Machine Learning Research*, 2020.

Jonas Freiburghaus, Aïcha Rizzotti-Kaddouri, and Fabrizio Albertetti. A deep learning approach for blood glucose prediction of type 1 diabetes. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA*, 2020.

Mufeng Geng, Xiangxi Meng, Jiangyuan Yu, Lei Zhu, Lujia Jin, Zhe Jiang, Bin Qiu, Hui Li, Hanjing Kong, Kun Yang, Hongming Shan, Hongbin Han, Zhi Yang, Qiushi Ren, and Yanye Lu. Content-noise complementary learning for medical image denoising. *IEEE Transactions on Medical Imaging*, 2022.

Lovedeep Gondara. Medical image denoising using convolutional denoising autoencoders. *IEEE 16th International Conference on Data Mining Workshops*, 2016.

GA Gregory, TIG Robinson, SE Linklater, F Wang, S Colagiuri, C de Beaufort, KC Donaghue, International Diabetes Federation Diabetes Atlas Type 1 Diabetes in Adults Special Interest Group;, DJ Magliano, J Maniam, TJ Orchard, P Rai, and GD Ogle GD. Global incidence, prevalence, and mortality of type 1 diabetes in 2021 with projection to 2040: a modelling study. *Lancet Diabetes Endocrinol.*, 2022.

Kang Gu, Ruoqi Dang, and Temiloluwa Prioleau. Neural physiological model: A simple module for blood glucose prediction. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2020.

Hadia Hameed and Samantha Kleinberg. Comparing machine learning techniques for blood glucose forecasting using free-living and patient generated data. *Proc Mach Learn Res.*, 2022.

Hadia Hameed and Samantha Klienberg. Investigating potentials and pitfalls of knowledge distillation across datasets for blood glucose forecasting. *International Workshop on Knowledge Discovery in Healthcare Data*, 2020.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *NeurIPS*, 2018.

James Arthur Harris and Francis Gano Benedict. A biometric study of basal metabolism in man. *Carnegie institution of Washington*, 1919.

Tobias Hatt, Jeroen Berrevoets, Alicia Curth, Stefan Feuerriegel, and Mihaela van der Schaar. Combining observational and randomized data for estimating heterogeneous treatment effects, 2022.

Jeremiah Hauth, Safa Jabri, Fahad Kamran, Eyoel W. Feleke, Kaleab Nigusie, Lauro V. Ojeda, Shirley Handelzalts, Linda Nyquist, Neil B. Alexander, Xun Huan, Jenna Wiens, and Kathleen H. Sienko. Automated loss-of-balance event identification in older adults at risk of falls during real-world walking using wearable inertial measurement units. *Sensors*, 2021.

Y Huang, S Ni, Z Lu, X He, J Hu, B Li, H Ya, and Y Shi Y. Heterogeneous temporal representation for diabetic blood glucose prediction, 2023.

Anqiang Huanga, Han Qiaob, and Shouyang Wang. Forecasting container throughputs with domain knowledge. *Procedia Computer Science*, 31, 2014.

Guido W Imbens. Nonparametric estimation of average treatment effects under exogeneity: A review. *Review of Economics and statistics*, 86(1):4–29, 2004.

A Janez, C Guja, A Mitrakou, N Lalic, T Tankova, L Czupryniak, AG Tabák, M Prazny, E Martinka, and L Smircic-Duvnjak L. Insulin therapy in adults with type 1 diabetes mellitus: a narrative review. *Diabetes Ther.*, 2020.

Insu Jeon, Wonkwang Lee, Myeongjang Pyeon, and Gunhee Kim. Ib-gan: Disentangled representation learning with information bottleneck generative adversarial networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:7926–7934, May 2021. doi: 10.1609/aaai.v35i9.16967. URL https://ojs.aaai.org/index.php/AAAI/article/view/16967.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. *ICML*, 2018.

Nathan Kallus, Aahlad Manas Puli, and Uri Shalit. Removing hidden confounding by experimental grounding, 2018.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel : Model-based offline reinforcement learning, 2021.

Kwanyoung Kim and Jong Chul Ye. Noise2score: Tweedie's approach to self-supervised image denoising without clean images. *NeurIPS*, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference for Learning Representations*, 2014.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2014.

Brent M. Kious, Amanda Bakian, Joan Zhao, Brian Mickey, Constance Guille, Perry Renshaw, and Srijan Sen. Altitude and risk of depression and anxiety: findings from the intern health study. *International Review of Psychiatry*, 2019.

Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void - learning denoising from single noisy images. *CVPR*, 2018.

Sören R Künzel, Jasjeet S Sekhon, Peter J Bickel, and Bin Yu. Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the National Academy of Sciences of the United States of America*, 116(10):4156, 2019.

Takuro Kutsuna. Supervised contrastive learning with heterogeneous similarity for distribution shifts, 2023.

Alex Labach, Aslesha Pokhrel, Seung Eun Yi, Saba Zuberi, Maksims Volkovs, and Rahul G Krishnan. Effective self-supervised transformers for sparse time series data, 2023. URL https://openreview.net/forum?id=HUCgU5EQluN.

Samuli Laine, Tero Karras, Jaakko Lehtinen, and Timo Aila. High-quality self-supervised deep image denoising. *NeurIPS*, 2019.

Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning, 2021.

Joyce M. Lee, Andrea Rusnak, Ashley Garrity, Emily Hirschfeld, Inas H. Thomas, Michelle Wichorek, Jung Eun Lee, Nicole A. Rioles, Osagie Ebekozien, and Sarah D. Corathers. Feasibility of Electronic Health Record Assessment of 6 Pediatric Type 1 Diabetes Self-management Habits and Their Association With Glycemic Outcomes. *JAMA Network Open*, 4(10):e2131278–e2131278, 10 2021. ISSN 2574-3805. doi: 10.1001/jamanetworkopen.2021.31278. URL https://doi.org/10.1001/jamanetworkopen.2021.31278.

Kimin Lee, Sukmin Yun, Kibok Lee, Honglak Lee, Bo Li, and Jinwoo Shin. Robust inference via generative classifiers for handling noisy labels. *ICML*, 2019.

Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5757–5766. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/lee20g.html.

Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. *ICML*, 2018.

You Lin, Juanhui Wang, and Mingjian Cui. Reconstruction of power system measurements based on enhanced denoising autoencoder. *IEEE General Meeting Power  Energy Society*, 2019.

Stan Lipovetsky and Michael Conklin. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry*, 17, 2001.

Loop Docs. Closed loop, 2023. URL https://loopkit.github.io/loopdocs/operation/loop/close-loop/.

Lalo Magni, Davide M. Raimondo, Luca Bossi, Chiara Dalla Man, Giuseppe De Nicolao, Boris Kovatchev, and Claudio Cobelli. Model predictive control of type 1 diabetes: An in silico trial. *Journal of diabetes science and technology*, 1, 2007.

Priyanka Majety. Insulin dosage: What do i need to know?, 2022.

Chiara Dalla Man, Francesco Micheletto, Dayu Lv, Marc Breton Boris Kovatchev, and Claudio Cobelli. The uva/padova type 1 diabetes simulator. *J Diabetes Sci Technol.*, 8, 2014.

Cindy Marling and Razvan C. Bunescu. The OhioT1DM dataset for blood glucose level prediction. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA*, 2018a.

Ciny Marling and Razvan C. Bunescu. The OhioT1DM dataset for blood glucose level prediction. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA*, 2018b.

Ciny Marling and Razvan C. Bunescu. The OhioT1DM dataset for blood glucose level prediction: Update 2020. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA*, 2020a.

Ciny Marling and Razvan C. Bunescu. The OhioT1DM dataset for blood glucose level prediction: Update 2020. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA*, 2020b.

Roger S. McIntyre, Zahinoor Ismail, Christopher P. Watling, Catherine Weiss, Stine R. Meehan, Primrose Musingarimi, and Michael E. Thase. Patient-reported outcome measures for life engagement in mental health: a systematic review. *Journal of Patient-Reported Outcomes*, 2022.

Stephen P McKenna. Measuring patient-reported outcomes: moving beyond misplaced common sense to hard science. *BMC Medicine*, 2011.

Richard McShinsky and Brandon Marsha. Comparison of forecasting algorithms for type 1 diabetic glucose prediction on 30 and 60-minute prediction horizons. *International Workshop on Knowledge Discovery in Healthcare Data*, 2020.

S.N. Mehta, N. Quinn, L.K. Volkening, and L.M. Laffel. Impact of carbohydrate counting on glycemic control in children with type 1 diabetes. *Diabetes Care*, 32, 2009.

Andrew C. Miller, Nicholas J. Foti, and Emily Fox. Learning insulin-glucose dynamics in the wild. *Machine Learning for Healthcare*, 126:1–25, 2020.

Sadegh Mirshekarian, Hui Shen, Razvan Bunescu, and Cindy Marling. Lstms and neural attention models for blood glucose prediction: Comparative experiments on real and synthetic data. *Annu Int Conf IEEE Eng Med Biol Soc.*, 2019.

Baharan Mirzasoleiman, Kaidi Cao, and Jure Leskovec. Coresets for robust training of deep neural networks against noisy labels. *NeurIPS*, 2020.

Nick Moran, Dan Schmidt, Yu Zhong, and Patrick Coady. Noisier2noise: Learning to denoise from unpaired noisy data. *CVPR*, 2019.

Mario Munoz-Organero. Deep physiological model for blood glucose prediction in t1dm patients. *Sensors*, 2020.

Aaron Neinstein, Jenise Wong, Howard Look, Brandon Arbiter, Kent Quirk, Steve McCanne, Yao Sun, Michael Blum, and Saleh Adi. A case study in open source innovation: developing the tidepool platform for interoperability in type 1 diabetes management. *Journal of the American Medical Informatics Association*, 23, 2016.

Hanh Nguyen, Phyllis Butow, Haryana Dhillon, and Puma Sundaresan. A review of the barriers to using patient-reported outcomes (pros) and patient-reported outcome measures (proms) in routine cancer care. *Journal of Medical Radiation Sciences*, 2020.

Pasquale Palumbo, Susanne Ditlevsen, Alessandro Bertuzzi, and Andrea De Gaetano. Mathematical modeling of the glucose–insulin system: A review. *Mathematical Biosciences*, 244, 2013.

Leonardos Pantiskas, Kees Verstoep, and Henri Bal. Interpretable multivariate time series forecasting with temporal attention convolutional neural networks. *IEEE Symposium Series on Computational Intelligence*, 2020.

Irfan Pratama, Adhistya Erna Permanasari, Igi Ardiyanto, and Rini Indrayani. A review of missing values handling methods on time-series data. *International Conference on Information Technology Systems and Innovation (ICITSI)*, 2016.

Yao Qin, Dongjin Song, Haifeng Cheng, Wei Cheng, Guofei Jiang, and Garrison W. Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. *Joint Conference on Artificial Intelligence (IJCAI)*, 2017.

Md Fazle Rabby, Yazhou Tu, Md Imran Hossen, Insup Lee, Anthony S. Maida, and Xiali Hei. Stacked lstm based deep recurrent neural network with kalman smoothing for blood glucose prediction. *BMC Medical Informatics and Decision Making*, 2021.

Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs Bergmann, and Roland Vollgraf and. Multi-variate probabilistic time series forecasting via conditioned normalizing flows. *International Conference on Learning Representations (ICLR)*, 2020.

Maria J. Redondo and Noel G. Morgan. Heterogeneity and endotypes in type 1 diabetes mellitus, 2023.

Zhongzheng Ren, Raymond Yeh, and Alexander Schwing. Not all unlabeled data are equal: Learning to weight data in semi-supervised learning. *NeurIPS*, 2020.

Peter J. Rockwell and Richard A. Davis. Multivariate time series. In Peter J. Rockwell and Richard A. Davis, editors, *Introduction to Time Series and Forecasting*, pages 227–257. Springer Texts, 2016.

Ignacio Rodriguez. Forecasting blood sugar levels in diabetes with univariate algorithms, 2021.

Paul R Rosenbaum and Donald B Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Donald B Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688, 1974.

H Rubin-Falcone, I Fox, E Hirschfeld, L Ang, R Pop-Busui, JM Lee, and J Wiens. Association between management of continuous subcutaneous basal insulin administration and hba1c. *J Diabetes Sci Technol.*, 2022.

Harry Rubin-Falcone, Ian Fox, and Jenna Wiens. Deep residual time-series forecasting: Application to blood glucose prediction. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA*, 2020.

Harry Rubin-Falcone, Joyce M. Lee, and Jenna Wiens. Forecasting with sparse but informative variables: A case study in predicting blood glucose. *AAAI*, 2023.

Viral N Shah, Lori M Laffel, R Paul Wadwa, and Satish K Garg. Performance of a factory-calibrated real-time continuous glucose monitoring system utilizing an automated sensor applicator. *Diabetes Technol Ther.*, 2018.

Uri Shalit, Fredrik D Johansson, and David Sontag. Estimating individual treatment effect: generalization bounds and algorithms. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3076–3085. JMLR. org, 2017.

Remei Calm Joaquim Armengol Silvia Oviedo, Josep Vehí. A review of personalized blood glucose prediction strategies for t1dm patients. *Int J Numer Method Biomed Eng*, 2016.

The Diabetes Control and Complications Trial Research Group. The effect of intensive treatment of diabetes on the development and progression of long-term complications in insulin-dependent diabetes mellitus. *N Engl J Med*, 1993.

Sindhu Tipirneni and Chandan K. Reddy. Self-supervised transformer for multivariate clinical time-series with missing values. *CoRR*, abs/2107.14293, 2021. URL https://arxiv.org/abs/2107.14293.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

Esmee M. van der Willik, Caroline B. Terwee, Willem Jan W. Bos, Marc H. Hemmelder, Kitty J. Jager, Carmine Zoccali, Friedo W. Dekker, and Yvette Meuleman. Patient-reported outcome measures ( proms): making sense of individual prom scores and changes in prom scores over time. *Nephrology*, 2020.

WPTM van Doorn, YD Foreman, NC Schaper, HHCM Savelberg, A Koster, CJH van der Kallen, A Wesselius, MT Schram RMA Henry, PC Dagnelie, BE de Galan, O Bekers, CDA Stehouwer, SJR Meex, and MCGJ Brouwers. Machine learning-based glucose prediction with use of continuous glucose and physical activity monitoring data: The maastricht study. *PLoS One*, 2021.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. *ICML*, 2008.

Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning, 2019.

Xin Wang, Hong Chen, Si'ao Tang, Zihao Wu, and Wenwu Zhu. Disentangled representation learning, 2022.

Xinzhi Wang, Huao Li, Hui Zhang, Michael Lewis, and Katia Sycara. Explanation of reinforcement learning model in dynamic multi-agent system, 2020.

Julia Ware, Janet M. Allen, Charlotte K. Boughton, Malgorzata E. Wilinska, Sara Hartnell, Ajay Thankamony, Carine de Beaufort, Ulrike Schierloh, Elke Fröhlich-Reiterer, Julia K. Mader, Thomas M. Kapellen, and Birgit Rami-Merhar. Randomized trial of closed-loop control in very young children with type 1 diabetes. *N Engl J Med*, 2022.

Priscilla Jia Ling Wee, Yu Heng Kwan, Dionne Hui Fang Loh, Jie Kie Phang, Troy H Puar, Truls Østbye, Julian Thumboo, Sungwon Yoon, and Lian Leng Low. Measurement properties of patient-reported outcome measures for diabetes: Systematic review. *Journal of Medical Internet Research*, 2021.

Ruth S Weinstock. Patient education: Type 1 diabetes: Insulin treatment (beyond the basics). *UpToDate*, 2023.

Pengxiang Wu, Songzhu Zheng, Mayank Goswami, Dimitris Metaxas, and Cho Chen. A topological filter for learning with label noise. *NeurIPS*, 2020.

Jinyu Xie. Simglucose v0.2.1 [online]. avaible: https://github.com/jxx123/simglucose. *Accessed on: Jan-20-2020*, 2018.

Yaochen Xie, Zhengyang Wang, and Shuiwang Ji. Noise2same: Optimizing a self-supervised bound for image denoising. *NeurIPS*, 2020.

Peng Xiong, Hongrui Wang, Ming Liu, Suiping Zhou, Zengguang Hou, and Xiuling Liu. Ecg signal enhancement based on improved denoising auto-encoder. *Engineering Applications of Artificial Intelligence*, 52, 2016.

Dongkuan Xu, Wei Cheng, Bo Zong, Dongjing Song, Jingchao Ni, Wenchao Yu, Yanchi Liu, Haifeng Chen, and Xiang Zhang1. Tensorized lstm with adaptive shared memory for learning trends in multivariate time series. *AAAI Conference on Artificial Intelligence*, 2020a.

Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline rl with no ood actions: In-sample learning via implicit value regularization, 2023.

Jian Xu. How to beat the cartpole game in 5 lines. *Towards Data Science*, 2021.

Jun Xu, Yuan Huang, Ming-Ming Cheng, Li Liu, Fan Zhu, Zhou Xu, and Ling Shao. Noisy-as-clean: Learning self-supervised denoising from the corrupted image. *TIP*, 2020b.

Taku Yamagata, Aisling O'Kane, Amid Ayobi, Dmitri Katz, Katarzyna Stawarz, Paul Marshall, Peter Flach, and Raúl Santos-Rodríguez. Model-based reinforcement learning for type 1diabetes blood glucose control, 2020.

J Yang, L Li, Y Shi, and X Xie. An arima model with adaptive orders for predicting blood glucose concentrations and hypoglycemia. *IEEE J Biomed Health Inform.*, 2019.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization, 2020.

Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor W. Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? *ICML*, 2019.

Haoming Zhang, Mingqi Zhao, Chen Wei, Dante Mantini, Zherui Li, and Quanying Liu. Eegdenoisenet: a benchmark dataset for deep learning solutions of eeg denoising. *Journal of Neural Engineering*, 2021.

Weijia Zhang, Lin Liu, and Jiuyong Li. Treatment effect estimation with disentangled latent factors. *arXiv preprint arXiv:2001.10652*, 2020.