Forecasting with Sparse but Informative Variables: A Case Study in Predicting Blood Glucose

Harry Rubin-Falcone
hrf@umich.edu
Division of Computer Science and Engineering, University of Michigan
Ann Arbor, MI, USA

Joyce M. Lee
joyclee@med.umich.edu
Division of Pediatric Endocrinology, University of Michigan
Ann Arbor, MI, USA

Jenna Wiens
wiensj@umich.edu
Division of Computer Science and Engineering, University of Michigan
Ann Arbor, MI, USA

Abstract

In time-series forecasting, future target values may be affected by both intrinsic and extrinsic effects. When forecasting blood glucose, for example, intrinsic effects can be inferred from the history of the target signal alone (*i.e.* blood glucose), but accurately modeling the impact of extrinsic effects requires auxiliary signals, like the amount of carbohydrates ingested. Standard forecasting techniques often assume that extrinsic and intrinsic effects vary at similar rates. However, when auxiliary signals are generated at a much lower frequency than the target variable (*e.g.*, blood glucose measurements are made every 5 minutes, while meals occur once every few hours), even well-known extrinsic effects (*e.g.*, carbohydrates increase blood glucose) may prove difficult to learn. To better utilize these *sparse but informative variables* (SIVs), we introduce a novel encoder/decoder forecasting approach that accurately learns the per-timepoint effect of the SIV, by (i) isolating it from intrinsic effects and (ii) restricting its learned effect based on domain knowledge. On simulated and real datasets pertaining to the task of blood glucose forecasting, our approach outperforms baseline approaches in terms of rMSE (Simulated: 13.07 [11.77,14.16] vs. 14.14, [95% CI: 12.69,15.27]; Real: 20.16 [19.28,21.06] vs. 20.36 [19.46,21.30]). We hypothesize that the smaller improvement observed in the Real dataset is due to noise and missingness in the SIV signal. By isolating their effects and incorporating domain knowledge, our approach makes it possible to better utilize noise-free SIVs in forecasting.

CCS Concepts

• **Computing methodologies** → **Neural networks**.

Keywords

time series, multi-input forecasting, diabetes, RNN

ACM Reference Format:
Harry Rubin-Falcone, Joyce M. Lee, and Jenna Wiens. 2020. Forecasting with Sparse but Informative Variables: A Case Study in Predicting Blood Glucose. In *KDD '22: Knowledge Discovery and Data Mining August 14–, 2022, Toronto, CA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/1122445.1122456

KDD '22, August 14–18, 2022, Washington DC

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

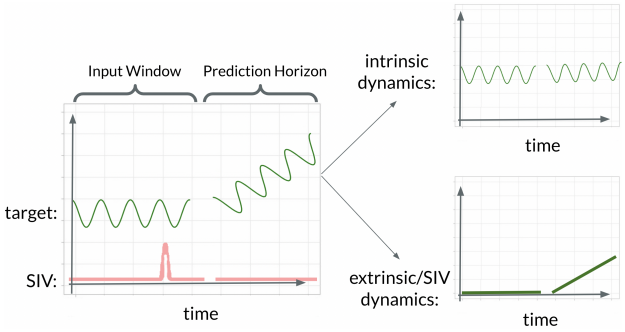

Figure 1: An overview of the SIV problem. In this toy example, the target variable exhibits oscillatory behavior when only zero SIV values are present (intrinsic dynamics), and the presence of a non-zero SIV value causes the target signal to increase linearly (extrinsic [SIV] dynamics).

1 Introduction

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 [2, 24]. 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 physiological variables like blood glucose. Auxiliary signals like 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 [9, 17, 25]. We hypothesize 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. A forecasting model that successfully addresses this problem will leverage the SIV despite its sparsity, leading to overall improved predictions.

1.1 Problem Definition

In this work, we introduce and address the SIV problem (**Figure 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. 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, but for most timepoints it is zero. While approaches for addressing missingness (or irregular sampling) have been extensively studied [21], the SIV problem has not.

1.2 Challenges

Although developing forecasting strategies that make use of SIVs has the potential to improve predictive accuracy in blood glucose forecasting and other domains, it has not been directly addressed in previous work. Recent multivariate forecasting approaches attempt to learn complex inter-variable dependencies [7, 22, 28]. However, these approaches do not explicitly account for the relative sparsity of some variables. Naive approaches to addressing the SIV issue include re-sampling the data so that more samples with non-zero SIV values are given to the network and carrying forward the SIV values to the end of the input window, but in practice we have found that these approaches generally fail to improve performance. 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.

1.3 Our Idea

To address the SIV problem we propose a novel forecasting approach. Our approach, "The Linked Encoder/Decoder", is inspired by the recursive nature of autoregressive models of the blood glucose system. 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.

1.4 Contributions

Our main contributions are summarized 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 through a novel linked encoder/decoder network.
- We evaluate our model on two blood glucose datasets pertaining to type 1 diabetes (T1D) and show that it more effectively incorporates SIVs compared to several baselines.

2 Problem Setup

Here we formalize our task and describe our motivating setting: blood glucose forecasting in type 1 diabetes.

2.1 Task Formalization

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 non-zero x' values has a known effect on the target signal (*e.g.*, they result 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: $\mathbf{y} = \mathbf{x}_{1:h}$.

As is common in forecasting work [3], we assume the target signal is generated by some underlying autoregressive process, and non-zero SIV values contribute in an additive autoregressive way. Note, our setup focuses on the setting in which extrinsic effects are driven by an SIV, but there are other settings where extrinsic effects may be driven by a variable that is not sparse, or a combination of sparse and non-sparse variables. We focus on the SIV problem and assume that any additional non-sparse extrinsic effects can be modelled using standard forecasting approaches.

2.2 A Motivating Example- Predicting Blood Glucose

The SIV problem arises in blood glucose forecasting, which has been extensively studied in the past [26], including in deep learning settings [5, 7, 19, 25]. Specifically, one aims to estimate blood glucose concentration (i.e., the target variable) for some prediction horizon into the future, based on a history of blood glucose and other signals. This represents a challenging forecasting task since glucose dynamics vary based on activity, time of day, hormone levels and more, resulting in significant non-stationarity throughout the day.

Accurate models for blood glucose forecasting are critical to the development of algorithms for managing blood glucose in individuals with diabetes (one in ten people in the US). Individuals with type one diabetes require insulin injections to maintain healthy glucose levels, throughout the day and especially around meals. Carbohydrates (i.e., meals) increase blood glucose, while insulin decreases blood glucose. However, current approaches do as well without information on carbohydrates or insulin as with [9, 17, 25].

In this setting, both insulin boluses and carbohydrates are considered sparse but informative variables or SIVs, since they occur only a few times a day. In contrast, blood glucose is recorded every 5-minutes. Carbohydrates and insulin result in an increase/decrease (respectively) of blood glucose after a delay of 30 minutes

to an hour. However, the effects of these variables are not always long lasting. As a result, blood glucose forecasters can learn to ignore these variables while still generating accurate predictions for most timepoints. However, models that utilize these variables will perform more accurately during critical changes in blood glucose.

3 Methods

Overview. To effectively capture the autoregressive dynamics of forecasting with an SIV, our architecture, the "Linked Encoder/Decoder", relies on a recursive framework (**Figure 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 signal is input directly into the SIV decoder. The two decoder systems are linked through a shared hidden state, which is processed in parallel, so that the intrinsic and extrinsic dynamics can be learned separately. Once isolated, we restrict the direction of the effect of the SIV on the target signal based on domain knowledge.

3.1 Architecture

Our Linked Encoder/Decoder contains two decoder systems that separately model intrinsic effects and extrinsic SIV effects. For model input windows that only contain zero SIV values, our architecture functions as a standard encoder/decoder.

Standard Encoder/Decoder. Our approach is based on a standard encoder/decoder recurrent neural network, as depicted by the orange and yellow sections of **Figure 2**. For samples where $\mathbf{x'}_{-T+1:0} = \mathbf{0}$, this encoder/decoder is not modified. A single encoder (ψ), 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 (θ) 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 θ, 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 $\hat{y}_t = FC(\mathbf{h}_{\theta t})$ for $t = 1, ...h$.

Linked SIV Decoder and Gating. For input samples that contain a non-zero SIV value, we augment this standard encoder/decoder with a second decoder ϕ that aims to model SIV dynamics, depicted in the blue section of **Figure 2**. By gating the output of the encoder based on the 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 θ, as described below. Because ϕ is only engaged when an SIV is present, θ learns to forecast in the absence of an SIV, while ϕ learns the effect of the SIV. The ψ network models combined effects.

SIV Decoder In order to encourage the SIV decoder to utilize the SIV, the decoder also receives the entire SIV signal as input. We shift the SIV signal at each timepoint so that the encoder's position in time relative to the SIV is included in the representation implicitly (see implementation details, section 4.4).

We incorporate knowledge regarding how the SIV affects the target variable in processing the output of ϕ at each time step. We pass $\mathbf{h}_{\phi t}$ (the output of ϕ) through a ReLU function after it is output by ϕ, before adding it to $\mathbf{h}_{\theta t}$ and passing the shared hidden state to subsequent time steps and the output network. If the SIV is expected to lead to a decrease in the target signal, $\mathbf{h}_{\phi t}$ is multiplied by -1 after it is passed through the ReLU function. This restricts the effect of the SIV on the target variable to the expected direction.

Linked Hidden State Processing. Both decoders process a single hidden state in parallel, and their outputs are summed, after restriction has been applied to the output of the SIV decoder. 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 (*i.e.*, as illustrated in Figure 2; we define: $\mathbf{h'}_{\theta t} = \mathbf{h}_{\theta t} + \mathbf{h}_{\phi t}$), for $t > 0$. This combined hidden state ($\mathbf{h'}_{\theta t}$) is passed to the output network (FC) and both decoders for the next step. The final forecast $\hat{\mathbf{y}}$ is a sum of the outputs of the two decoders capturing both intrinsic and restricted extrinsic effects (*i.e.*, $\hat{y}_t = FC(\mathbf{h'}_{\theta t})$). Note that when $\mathbf{x'} = \mathbf{0}$, ϕ is not engaged, and $\mathbf{h'}_{\theta t} = \mathbf{h}_{\theta t}$.

Additional Variables. In **Figure 2** we present an overview of the proposed architecture for a setting with a single SIV. In the setting of multiple SIVs, one would increase the number of secondary decoders and apply restrictions according to the known effect of each SIV. Each ϕ would take 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. In addition, non-sparse auxiliary variables, if any, are given to the ψ network, along with $\mathbf{x}$ and $\mathbf{x'}$, so that non-SIV extrinsic effects can be modeled (by θ, since these variables do not effect gating). The θ network must perform well in the absence of non-zero SIV signals, thus the ϕ network is encouraged to learn the remaining extrinsic effects for samples with non-zero SIV values (which must be the SIV effects).

3.2 SIV Representation

One issue that makes utilizing SIVs difficult is that they usually occur at only one timepoint in the input window, having little effect on gradient calculations. To increase its effect, we use the sum-total SIV value up to the current timepoint as input (**Figure 3**). Up until the first non-zero SIV value of an input time-series, the signal 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 was found to improve performance during baseline tuning and is used for all analyses reported here, including baselines and ablations. We evaluate the impact of this approach in Appendix B.

4 Experimental Setup

We evaluate our approach on two datasets pertaining to blood glucose forecasting in T1D, which serves as our motivating example of the SIV problem. We compare to several relevant baselines, including common resampling approaches. We evaluate forecast accuracy and the extent to which each model utilizes the SIVs.

4.1 Forecasting Task

We aim to forecast blood glucose values 30 minutes into the future ($h = 6$), based on a history of blood glucose and two sparse but informative variables: carbohydrate and insulin bolus values from the past 2 hours ($T = 24$). Here, $h = 6$ as it represents a common BG

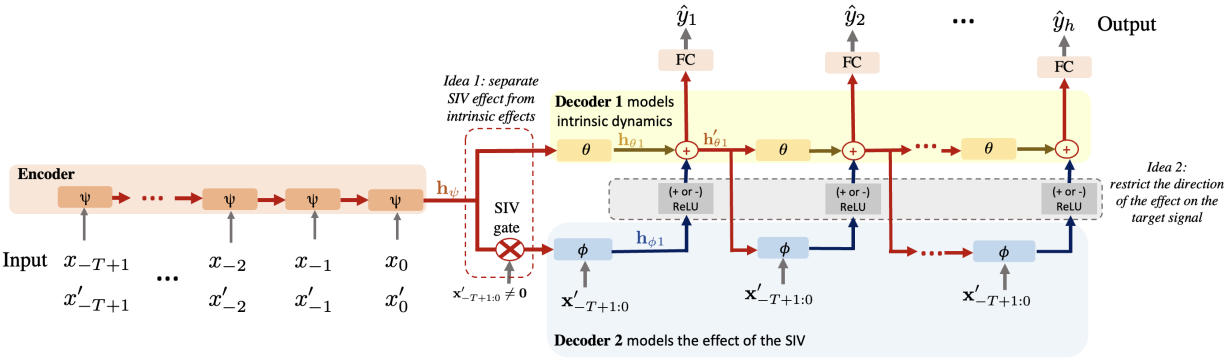

Figure 2: Our model's architecture: the Linked Encoder/Decoder, shown with an input length T and prediction horizon h. The θ network models intrinsic dynamics, while the ϕ network models the SIV dynamics. The ψ network models shared dynamics. Input time-series are gated, such that only inputs ($[\mathbf{x}_{-T+1:0}, \mathbf{x}'_{-T+1:0}]$) containing a non-zero SIV at any timepoint are passed through the ϕ network. The separation between intrinsic and extrinsic dynamics is used to ensure that the relationship between SIV and target is as expected via the ReLU network shown in grey.

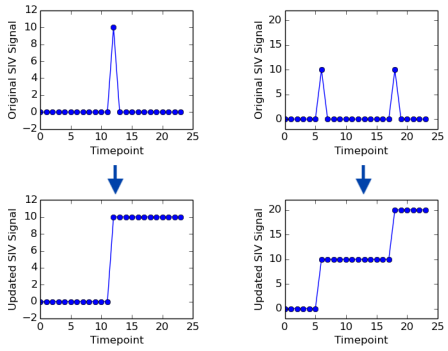

Figure 3: Our sum-total approach. We use the sum total up to the current point within an input window as input. This method allows the SIV signal to make a larger impact on the gradient while maintaining all temporal information.

forecasting benchmark [15] and we set T=24 based on prior work that suggests longer histories do not provide additional benefit [26].

4.2 Datasets

We compare the performance of our architecture to baselines on two T1D-based datasets: Simulated and Real. Both datasets are publicly available and have been previously explored in the context of forecasting [14, 15, 27]. Each dataset consists of blood glucose, bolus (fast-acting) insulin, basal (slow-acting) insulin, carbohydrate values, and sine and cosine of time of day (to capture non-stationarity) for real or simulated individuals with T1D. Bolus insulin and carbohydrate values are considered SIVs. All variables were scaled to be between zero and one. Each dataset was split into overlapping windows of length $T + h$ with a stride of 1, to be used as model input and ground truth values.

Simulated. Data generated from a commonly-used T1D simulator provide a curated test setting with no missingness or human measurement error on which to evaluate our approach. We used the UVA-Padova simulator [14], via a publicly available implementation [27]. We generated ten days of data for ten individuals (the ten "adult" patients modeled in the simulator), corresponding to 28,800 timepoints. Carbohydrate and insulin values occurred every 111 timepoints on average (Carbohydrates median, [IQR]: 84 timepoints between occurrences, [60,150]. Boluses: 83, [58,148]). The meal schedule used to generate simulated data was based on the Harrison-Benedict equation [10] as implemented in [6], but without snacks (3 meals a day), to further highlight the SIV problem. In our simulation, we used the default basal-bolus controller from the existing implementation to administer insulin, but we delayed three quarters (randomly selected) of the bolus administrations to be 20 minutes to two hours after the simulated meal, randomly sampled from a uniform distribution. This delay was to control for a separate issue that can arise when multiple SIVs occur at precisely the same time. The delay disentangles their effects.

Real. This dataset includes both the OHIOT1DM 2018 and 2020 datasets, developed for the Knowledge Discovery in Healthcare Data Blood Glucose Level Predication Challenge [16]. 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.

4.3 Baselines

Encoder/Decoder. Our primary baseline is a stand-alone encoder/decoder system, identical to the ψ plus θ networks in our full architecture [5]. Due to the smaller capacity compared to our proposed approach, we increase the minimum number of training iterations so that we match the same number of gradient updates as our complete approach. We also examine a higher capacity model.

High 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 ϕ, and no output restriction).

Resampling. Resampling is perhaps the simplest common-sense approach to addressing signal sparsity. In order to rule out re-sampling as a naive solution to the SIV problem, we implement our primary baseline method with two re-sampling 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 our complete method's number of gradient updates.

4.4 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 ϕ 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 ϕ network while maintaining time information, $\mathbf{x}'_{-T+1:0}$ is front-padded with h zeros and input to ϕ 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.

4.5 Training Details

We split each dataset into training, validation and test sets used for evaluation purposes. For the simulated dataset we evaluated on the last 15% of data points, and for the real dataset we evaluated on the held-out test data from the challenges. For both datasets, the remaining data were split into 80% train and 20% validation. We implemented and trained 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 [12] and a batch size of 500. We used 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 was used as a loss function. Models were trained for at least 25 epochs, and then until validation data performance did not improve for 10 epochs. For simulated glucose, due to a smaller sample size, we trained for at least 500 epochs, until performance did not improve for 50 epochs. Parameters found at the iteration for which the model performed best on the validation data were used at inference time. For both datasets, we train and test a model on each subject (10 for Simulated, 12 for Real), and report across-subject averages.

4.6 Evaluation

All evaluations were performed on held-out test sets. In all comparisons, we measured forecasting performance using rMSE and mean absolute error (MAE). In order to match common practice in the blood glucose forecasting literature, we calculated error terms based on the prediction accuracy of the final timepoint in the prediction window [15]. We still train using MSE across all forecast horizon timepoints, as we find this added supervision to be generally helpful. We specifically probed the predictions to characterize to what extent they relied on the SIVs, by examining their error when forecasting without access to those variables.

SIV Usage metric. 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 $L(f_\emptyset(\mathbb{X}_\emptyset))$ - $L(f(\mathbb{X}))$. It is inspired by the Shapley Regression Value [13]. This metric reflects how error changes when the SIV is 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.

Effect of SIV Magnitude and Timing. We examine the SIV usage of our model vs the baseline encoder/decoder for input samples that contain non-zero SIV values vs those that do not, and also examine only windows that contain a carbohydrate value above the 50^{th} and 90^{th} percentiles, including windows from all individuals. We perform this experiment to validate that our metric and model are sensitive to the presence of SIVs generally, and highly impactful SIVs specifically. Because boluses and carbohydrates have a delayed transient effect, we also examine SIV usage for the baseline encoder/decoder and our proposed approach for windows whose final non-zero SIV value occurs in the first, second, third, and fourth half hours of the two hour input window. We hypothesize that SIV usage will be higher when non-zero SIV values occur later in the input window, because the inclusion of an SIV signal should be most beneficial when the SIV impacts the forecast horizon without having time to impact the target signal input.

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 in both datasets. 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. Finally, we perform the following ablation analyses to examine which elements most contribute to our models performance:

(1) **No Gating**: All samples are passed through both decoders.
(2) **No Restriction**: The outputs of the SIV decoder systems are not passed through a ReLU function.
(3) **No Dec. SIV Input**: The Decoder receives only the hidden state as input, and not the SIV signal.
(4) **Only Dec. SIV Input**: The baseline model is used, but with the modification that the SIV is input to the decoder directly, as in our model.

4.7 Data and Code Availability

All code, the simulated dataset, and the appendix are publicly available: https://gitlab.eecs.umich.edu/mld3/sparse-informative-variables.

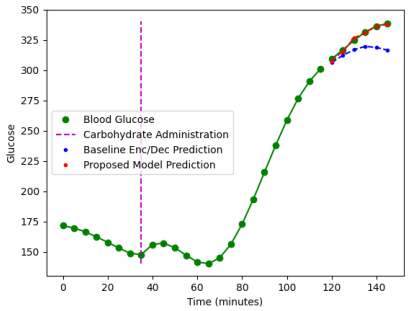

Figure 4: A sample prediction for a simulated individual (adult#004). Our model better accounts for the steep rise in the blood glucose signal following a meal.

The Real 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.

5 Results and Discussion

We compare our approach to several baselines across two datasets. We aim to answer the following questions.

- Does our model offer improvement over baseline approaches in terms of forecast error and SIV Usage? (Section 5.1)
- How does SIV usage and model performance vary with regard to SIV magnitude and timing? (Section 5.2)
- Across individuals, when does our model offer the greatest improvement? (Section 5.3)
- What elements of our model improve forecast accuracy and SIV usage? (Section 5.4)
- How is our model impacted by SIV missingness and noise? (Section 5.5)

5.1 Improvement Over Baselines

Our model outperforms baselines across both datasets, leading to lower rMSE/MAE and greater relative SIV usage (**Table 1**). Our model is better able to account for the effect of the SIV on the target signal. For example, accurately predicting sharp rises that the baseline encoder/decoder is unable to fully account for (**Figure 4** shows an example forecast that was selected for illustrating this phenomenon). Also of note, the naive SIV re-sampling approaches are generally outperformed by other baseline approaches, or perform similarly. For the simulated blood glucose dataset, our approach shows a large improvement over baseline (rMSE 13.07 vs 14.14). On the Real dataset, performance gains are more moderate (rMSE 20.16 vs 20.36). In general, the increased SIV usage of our proposed approach corresponds to lower error, demonstrating the informativeness of our approach. Multiple approaches exhibit negative SIV usage for the Real dataset, indicating that including the SIVs does more harm then good. We hypothesize that this is due to noise in the carbohydrate signal (explored further in section 5.5).

5.2 Input Level Results

We do not expect the benefit of our proposed approach to be constant across all settings. In particular, when the magnitude of the SIV is small or the SIV occurs early on in the prediction window, a model that ignores the SIV might still generate accurate forecasts. Thus, we evaluate differences between our approach and the encoder/decoder baseline as the magnitude and timing of the SIV varies.

Table 1: Forecasting Error and SIV usage for all datasets. Outcomes are reported as: Error [95% confidence interval] (SIV Usage). Our proposed approach outperforms baseline for both datasets. Confidence intervals were calculated from bootstraps with 1,000 re-samples.

Model	rMSE [95%CI] (Usage)	MAE [95%CI] (Usage)
	Simulated	
Encoder/Decoder	15.63,[14.08,16.89] (11.13)	12.42,[11.14,13.59] (6.63)
SIV Fine-tune	27.30,[24.66,29.09] (-0.54)	22.22,[19.9,23.89] (-3.17)
SIV Initialize	15.37,[13.63,16.86] (11.39)	11.99,[10.68,13.17] (7.06)
High Capacity	14.14,[12.69,15.27] (12.62)	11.21,[9.97,12.24] (7.85)
Proposed	**13.07,[11.77,14.16] (13.69)**	**10.45,[9.37,11.37] (8.61)**
	Real	
Encoder/Decoder	20.36,[19.46,21.30] (0.08)	14.67,[14.11,15.24] (0.24)
SIV Fine-tune	21.74,[20.87,22.64] (-1.30)	16.25,[15.68,16.85] (-1.35)
SIV Initialize	20.98,[20.00,21.97] (-0.54)	14.99,[14.42,15.59] (-0.09)
High Capacity	20.98,[20.04,21.92] (-0.54)	15.09,[14.5,15.69] (-0.18)
Proposed	**20.16,[19.28,21.06] (0.28)**	**14.64,[14.09,15.20] (0.27)**

Comparing the performance of the baseline and the proposed approach on windows that do not contain SIV vs those that do (Min. Carb. Percentile >0), performance gains over the baseline increase when SIV are present (**Figure 5** (a), left most box plots). Furthermore, SIV usage of both models increases as a function of carbohydrate value (*i.e.*, magnitude), indicating that larger values more greatly impact forecast error, as expected.

In our examination of SIV usage and SIV position within the input window, usage peaks in the third quartile (i.e., between 30 and 60 minutes prior to the prediction horizon, **Figure 5** (b)). This is the point at which carbohydrates and boluses most effect the prediction window, given their delay in effects and the length of the prediction horizon. If a non-zero SIV value occurs early enough in the window, its information is encoded in the input signal prior to the forecast horizon. This is why usage is low for windows with first quartile SIVs; the effect of the SIV is already captured by the target signal and the SIV can be ignored. Usage is highest in the third, rather than fourth, quartile, likely because the 30 to 60 minute delay in SIV effect is such that fourth quartile SIVs have not fully influenced the target value at the time of the forecast horizon. These trends hold to a certain extent on the Real data. However, the difference is less pronounced given the overall difficulty of the forecasting task with real data (Appendix A, **Figure 8**).

5.3 Individual Level Results

In our simulated results, we consider ten different individuals who differ in terms of meal schedules and simulated physiological parameters. Here, we investigate how model performance with respect to the baseline (i.e., the Encoder/Decoder) varies across these ten individuals. And in particular, we identify trends that predict settings in which our approach is more beneficial.

For both datasets, our model's benefit over the baseline encoder/decoder varies inversely with the extent to which the baseline

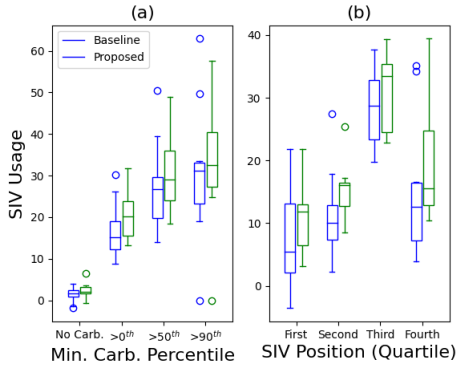

Figure 5: (a) Our architecture and baseline encoder/decoder SIV usage for windows without/with carbohydrates, and windows with carbohydrates above the 50^{th} and 90^{th} percentiles (Simulated dataset). Usage increases when larger carbs are present. (b) SIV usage for windows who's final non-zero SIV value occur in each quartile of the input time series (Simulated data). Usage peaks in the third quartile, which is the point at which carbohydrates and boluses most effect the prediction window.

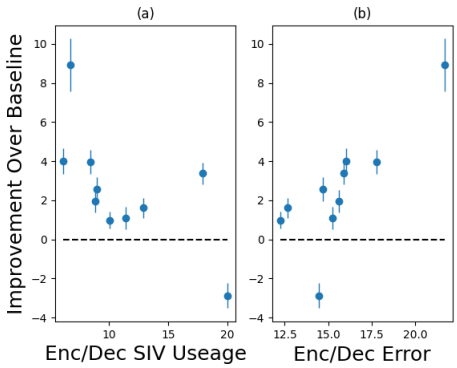

Figure 6: (a) Our architecture's improvement over the encoder/decoder baseline vs baseline SIV usage for the Simulated dataset. Our method's benefit increases as baseline SIV usage decreases. (b) Improvement over baseline vs baseline prediction error for Simulated data, for each individual in the simulated dataset. Improvement over baseline is positively correlated with baseline error, indicating that our model addresses per-subject deficits in the baseline approach.

approach relies on the SIV (*i.e.*, SIV usage) across individuals (Simulated: Pearson r=-0.65, p=0.041 [**Figure 6 (a)**], Real: Pearson r=-0.59, p=0.042, [Appendix A **Figure 9 (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, while individuals with low usage stood to benefit.

We also observe a strong correlation between baseline error and our approach's improvement (r=0.80, p= 0.0056, **Figure 6 (b)**). This suggests that our approach addresses the deficits of the baseline at the individual level, decreasing variation in the error across individuals. The higher variability in the performance of the baseline across individuals compared to the proposed approach (range: 6.3 vs 9.5) may be due in part to difficulties in SIV modeling, which our model is able to compensate for. With respect to the real data, while the overall trend was the same, the correlation between baseline error and our approach's improvement over baseline was not significant (r=0.22, p=0.49, Appendix A **Figure 9 (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 Real dataset may simply be more complex and thus result in more variability across individuals.

5.4 Ablations

To better understand the benefit of each part of our approach, we run ablations on both the simulated and real datasets. **Simulated dataset.** Ablation analyses reveal that, in general, our approach's strong SIV usage and forecast accuracy are a combined effect of each implementation detail taken together, rather than only one component. **Table 2** shows the results of our ablation study on the simulated dataset: removing any component results in a decrease in performance accuracy and SIV usage for all metrics. For the

Table 2: rMSE and MAE, with SIV usage, for each ablation on the Simulated dataset. Outcomes are reported as: Error [95% confidence interval] (SIV Usage).

Model	rMSE [95%CI] (Usage)	MAE [95%CI] (Usage)
No Gating	13.93,[12.57,15.04] (12.84)	11.11,[9.94,12.11] (7.95)
No Restriction	13.12,[11.8,14.21] (13.64)	10.43,[9.31,11.38] (8.62)
No Dec. SIV Input	14.20,[12.67,15.36] (12.56)	11.18,[9.96,12.23] (7.87)
Only Dec. SIV Input	13.97,[12.58,15.10] (12.79)	11.12,[9.96,12.15] (7.94)
Full Model	13.07,[11.77,14.16] (13.69)	10.45,[9.37,11.37] (8.61)

Simulated dataset, we see that our model performs similarly when the restriction is removed, *i.e.* when the ReLU functions are not included. This is likely because, in the Simulated dataset, the effect of the insulin boluses and carbohydrate administrations are clear enough that the model can learn them easily without supervision.

Real dataset. The restriction element is important for the Real dataset (Appendix A **Table 3**, rMSE increases to 20.38 from 20.16 when restriction is removed), which presents a more difficult challenge due to noise in the SIV signal and more complex target variable dynamics. For the real 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 Real 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.

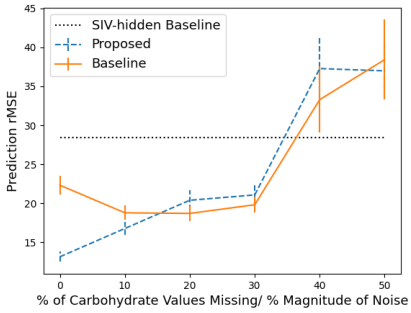

Figure 7: Baseline and proposed performance as simulated carbohydrate values are hidden from the model and noise is added to their magnitudes. As noise increases, our model's advantage disappears.

5.5 Sensitivity Analyses: Missingness and Noise

We hypothesize that the smaller gains on the real dataset may be due in part to noise in the measurement of the SIV. We use simulated data to examine this hypothesis, because in this dataset we have ground truth carbohydrate values. In the real data not only may the magnitude of these values be inaccurate (they are estimated by the patient), but the timing might also be off. Moreover, patients may skip recordings altogether. In order to examine the hypothesis that unreliable carbohydrate values hamper our model's performance on real data, we randomly hide between 10% and 50% of the carbohydrate values (by setting their value to zero) and add between 10% and 50% magnitude random noise to the remaining non-zero carbohydrate measurements (both after data generation, and to both training and testing datasets), to evaluate the performance of our model vs the stand-alone encoder/decoder baseline as carbohydrate values become unreliable. We chose a simulated individual for which our method performed strongly (adult#010) for this analysis.

We find that as the missingness and noise increase, our approach's performance degrades (**Figure 7**). Our approach is more impacted by missing or noisy SIV values relative to the baseline, in part because of the increased dependence on the SIV (*i.e.*, greater SIV usage). When 20% of carbohydrates are missing and the remaining carbohydrate values have 20% random noise added, our model no longer offers an improvement over baseline, and in fact performs slightly worse. We also see that SIV usage drastically decreases as the carbohydrate signal becomes unreliable. The model trained and tested without insulin and carbohydrates is not be effected by the noise, so baseline SIV usage is a constant minus baseline performance across missingness and noise values. Once the noise reaches 40% missing values with 40% added noise, performance of both approaches becomes worse than the SIV-hidden baseline, which means that the models have negative usage values. This indicates that including SIVs in the model can cause more harm than good if they are noisy enough.

6 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. The SIV problem frequently arises in healthcare, and SIVs are often 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. 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. However, each has limitations in an SIV setting as discussed below, and thus are not directly comparable.

Notably, Pantiskas et al. and Qin et al. use attention mechanisms to identify which variables to focus on [20, 22], but these approaches do not account for signals that are mostly zero-valued, nor do they incorporate domain knowledge as our approach does. In a probabilistic setting, normalizing flows have been used to directly model the joint probabilities between variables [4, 23]. This approach does not address the SIV setting because SIVs are often too sparse to accurately estimate a joint probability. Several approaches are based on using convolutional neural networks (CNNs) to process input variables prior to using a downstream forecaster [7, 8, 20], while other approaches attempt to model the relationship between variables via correlation graph inputs [1], multitask prediction across variables with a shared hidden state [28], or separate networks for each variable which are concatenated downstream [19]. Although each of these approaches explicitly models inter-variable relationships, none explicitly address the sparsity issue, which our architecture was designed to overcome. While some of these architectures isolate inter-variable effects, none use this isolation to restrict model outputs to match expectations. In contrast, we used isolation to inject domain knowledge by putting restrictions on the SIV effect.

Previous work in forecasting has combined deep learning with domain knowledge to reduce the hypothesis space. However, authors have relied on strong assumptions, *e.g.* structuring deep architectures to match clinical intuition [19], combining deep approaches with physiological-model-based simulators [18], and estimating expert judgements on model outputs via Monte-Carlo approximations [11]. In contrast, we only restrict the sign of the SIV network's hidden state.

7 Conclusions

The SIV problem arises in forecasting domains when the relative sparsity of an auxiliary signal makes it challenging to learn its effect on a target signal. Here, 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 than baselines and much stronger forecasting performance on simulated data. On real data, our model offers only modest improvements in forecasting error due to noise in the SIV signal. However, ablation results show that each aspect of our approach is beneficial, even in this noisy setting. Future work could work on first reducing input noise, which would potentially enable our model to perform well in the real data setting.

While there are many different ways to forecast signals, we focus on a family of RNN-based techniques. Our primary contribution is the identification of the Sparse but Informative variable (SIV) problem in forecasting, and the failure of common RNN-based approaches to address 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 instead focus on advancements to a specific family of techniques. Our findings could apply to many settings in which variants of RNNs are applied to forecasting problems with SIVs. Our approach to addressing the SIV problem involves gating, output restriction, and inputting variables directly to the decoder. While none 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.

Our study is not without limitations. First, confidence intervals overlapped in many of our experiments. This is likely due to the small number of datapoints with non-zero carbohydrate and bolus values. Although differences were small, many were consistent across patients and datasets. Second, our approach involved splitting the data into discrete windows in order to train using stochastic gradient descent. A problem with this approach is that any SIV values that appear before the input window, even only one time step before, are ignored. This means that windows that are assumed to not contain an SIV can still be influenced by them. If our approach was to be rolled out for real time patient blood-glucose prediction, for example, continuous acquisition and prediction could alleviate this problem by allowing SIV effects to go past the input window length. Although our approach is limited by this explicit fixed SIV effect time window and an inability to address noisy SIV signals, our model shows improvement over baseline performance in prediction error. This suggests that SIV effects can be better incorporated into a forecasting model, provided that domain knowledge is adequately leveraged.

Our approach was designed with blood glucose forecasting in mind, but does not directly address all problems that arise in this domain. If more than one SIV are present (as in blood glucose forecasting), our approach relies on having some non-overlapping SIV signals (*i.e.*, there must be timepoints where one SIV occurs, but not the other). This is limiting in some simulated settings, however, in real patient data, we can expect to see many non-overlapping signals because bolus administrations often follow carbohydrate consumption by an hour or more. We also do not address missing glucose signals in this work, which is a common issue in real data. Generally, and in blood glucose forecasting specifically, an SIV measurement early in the input window may have less of an effect on the target variable at the time of forecast. This is learned by ϕ, which can learn a decay effect over time.

While we restrict our analysis to blood glucose forecasting in T1D, the SIV problem is potentially relevant whenever an informative auxiliary signal is mostly zeros (or some other constant). The findings here could be applied more broadly to other domains, *e.g.* blood pressure prediction, where the SIV is vassopressor/vasodilator administration; the prediction of other vital signs, where medication administration represents an SIV; stock prediction, where quarterly reports are produced much less frequently than stock valuations; and travel, where holidays and major events are SIVs.

References

[1] Defu Cao, Yujing Wang1, Juanyong Duan, Ce Zhang, Xia Zhu, Conguri Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. 2020. Spectral Temporal Graph Neural Network for Multivariate Time-series Forecasting. *Conference on Neural Information Processing Systems (NeurIPS)* (2020).

[2] Kanad Chakraborty, Kishan Mehortra, Chilukuri Mohan, and Sanjay Ranka. 1992. Forecasting the behavior of multivariate time series using neural networks. *Neural Networks* 5 (1992), 961–970. Issue 6.

[3] Chris Chatfield. 2000. In *Time-Series Forecasting*. Chapman Hall/CRC.

[4] Syama Sundar Rangapuram Emmanuel de Bezenca an, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle3, Lorenzo Stella Hilaf Hasson, Patrick Gallinari, and Tim Januschowski. 2020. Normalizing Kalman Filters for Multivariate Time Series Analysis. *Conference on Neural Information Processing Systems (NeurIPS)* (2020).

[5] Ian Fox, Lynn Ang, Mamta Jaiswal, Rodica Pop-Busui, and Jenna Wiens. 2018. Deep Multi-Output Forecasting. *KDD* (2018).

[6] Ian Fox, Joyce Lee, Rodia Pop-Busui, and Jenna Wiens. 2020. Deep Reinforcement Learning for Closed-Loop Blood Glucose Control. *Proceedings of Machine Learning Research* (2020).

[7] Jonas Freiburghaus, Aïcha Rizzotti-Kaddouri, and Fabrizio Albertetti. 2020. A Deep Learning Approach for Blood Glucose Prediction of Type 1 Diabetes. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA* (2020).

[8] Kang Gu, Ruoqi Dang, and Temiloluwa Prioleau. 2020. Neural Physiological Model: A Simple Module for Blood Glucose Prediction. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (2020).

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

[10] James Arthur Harris and Francis Gano Benedict. 1919. A biometric study of basal metabolism in man. *Carnegie institution of Washington* (1919).

[11] Anqiang Huanga, Han Qiaob, and Shouyang Wang. 2014. Forecasting Container Throughputs With Domain Knowledge. *Procedia Computer Science* 31 (2014).

[12] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *International Conference for Learning Representations* (2014).

[13] Stan Lipovetsky and Michael Conklin. 2001. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry* 17 (2001). Issue 4.

[14] Chiara Dalla Man, Francesco Micheletto, Dayu Lv, Marc Breton Boris Kovatchev, and Claudio Cobelli. 2014. The UVA/PADOVA Type 1 Diabetes Simulator. *J Diabetes Sci Technol.* 8 (2014). Issue 1.

[15] Cindy Marling and Razvan C. Bunescu. 2018. The OhioT1DM Dataset for Blood Glucose Level Prediction. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA* (2018).

[16] Ciny Marling and Razvan C. Bunescu. 2020. The OhioT1DM Dataset for Blood Glucose Level Prediction: Update 2020. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA* (2020).

[17] Richard McShinsky and Brandon Marsha. 2020. 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).

[18] Andrew C. Miller, Nicholas J. Foti, and Emily Fox. 2020. Learning Insulin-Glucose Dynamics in the Wild. *Machine Learning for Healthcare* 126 (2020), 1–25.

[19] Mario Munoz-Organero. 2020. Deep Physiological Model for Blood Glucose Prediction in T1DM Patients. *Sensors* (2020).

[20] Leonardos Pantiskas, Kees Verstoep, and Henri Bal. 2020. Interpretable Multivariate Time Series Forecasting with Temporal Attention Convolutional Neural Networks. *IEEE Symposium Series on Computational Intelligence* (2020).

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

[22] Yao Qin, Dongjin Song, Haifeng Cheng, Wei Cheng, Guofei Jiang, and Garrison W. Cottrell. 2017. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. *Joint Conference on Artificial Intelligence (IJCAI)* (2017).

[23] Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs Bergmann, and Roland Vollgraf and. 2020. Multi-variate Probabilistic Time Series Forecasting via Conditioned Normalizing Flows. *International Conference on Learning Representations (ICLR)* (2020).

[24] Peter J. Rockwell and Richard A. Davis. 2016. Multivariate Time Series. In *Introduction to Time Series and Forecasting*, Peter J. Rockwell and Richard A. Davis (Eds.). Springer Texts, 227–257.

[25] Harry Rubin-Falcone, Ian Fox, and Jenna Wiens. 2020. Deep Residual Time-Series Forecasting: Application to Blood Glucose Prediction. *International Workshop on Knowledge Discovery in Healthcare Data-KHD@IJCA* (2020).

[26] Remei Calm Joaquim Armengol Silvia Oviedo, Josep Vehí. 2016. A review of personalized blood glucose prediction strategies for T1DM patients. *Int J Numer Method Biomed Eng* (2016).

[27] Jinyu Xie. 2018. Simglucose v0.2.1 [Online]. Avaible: https://github.com/jxx123/simglucose. *Accessed on: Jan-20-2020* (2018).

[28] Dongkuan Xu, Wei Cheng, Bo Zong, Dongjing Song, Jingchao Ni, Wenchao Yu, Yanchi Liu, Haifeng Chen, and Xiang Zhang1. 2020. Tensorized LSTM with Adaptive Shared Memory for Learning Trends in Multivariate Time Series. *AAAI Conference on Artificial Intelligence* (2020).

A Additional Results for Real Dataset

We observed that our method's performance gains over baseline for the Real dataset were more moderate on the Simulated dataset, perhaps due to noise in the carbohydrate signal, as explored in section 5.5. We report results on the Real dataset here, where we see similar trends as observed in the simulated data, but to a lesser degree. **Figure 8** shows how SIV usage varies as a function of SIV magnitude and position. **Figure 9** depicts how improvements over baseline vary across individuals, and **Table 3** contains ablation results for the Real dataset.

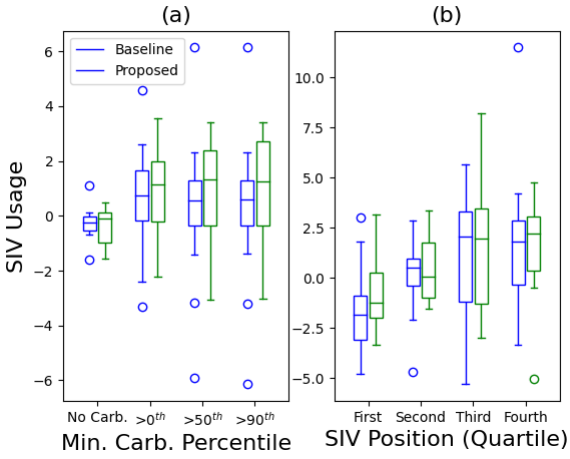

Figure 8: (a) Our architecture and baseline encoder/decoder SIV usage for windows without carbohydrates, windows with any carbohydrates, and windows with carbohydrates above the 50^{th} and 90^{th} percentiles for the Real dataset, all individuals. We see that in general usage and improvement over baseline increase when larger carbs are present. (b) SIV usage for windows who's final non-zero SIV value occur in the first, second, third and fourth quartiles of the input time series for Real data, all individuals. Usage peaks in the third quartile, which is the point at which carbohydrates and boluses most effect the prediction window, given their delay in effects. This effect is less pronounced than in the Simulated dataset.

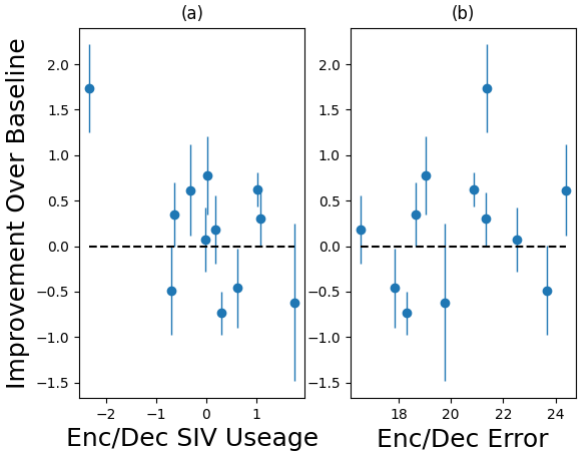

Figure 9: (a) Our architecture's improvement over the encoder/decoder baseline vs baseline SIV usage for the Real dataset. Our method's benefit increases as baseline SIV usage decreases. (b) Improvement over baseline vs baseline prediction error for Real data, for each individual in the simulated dataset. Improvement over baseline is not correlated with baseline error.

Table 3: 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 re-samples.

Model	rMSE [95%CI] (Usage)	MAE [95%CI] (Usage)
	Real	
No Gating	20.34,[19.48,21.27] (0.11)	14.77,[14.22,15.33] (0.13)
No Restriction	20.38,[19.48,21.31] (0.06)	14.73,[14.16,15.32] (0.18)
No Dec. SIV Input	20.71,[19.81,21.64] (-0.27)	14.97,[14.41,15.55] (-0.07)
Only Dec. SIV Input	20.52,[19.62,21.47] (-0.08)	14.70,[14.14,15.29] (0.20)
Proposed	20.16,[19.28,21.06] (0.28)	14.64,[14.09,15.20] (0.27)

B 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 (**Figure 4**).

Table 4: 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 re-samples.

Model	rMSE [95%CI] (Usage)	MAE [95%CI] (Usage)
	Simulated- Carry Forward	
Encoder/Decoder	15.63,[14.08,16.89] (11.13)	12.42,[11.14,13.59] (6.63)
Proposed	**13.07,[11.77,14.16] (13.69)**	**10.45,[9.37,11.37] (8.61)**
	Simulated- NO Carry Forward	
Encoder/Decoder	16.46,[14.64,17.84] (10.30)	12.97,[11.53,14.13] (6.09)
Proposed	16.08,[14.46,17.37] (10.68)	12.80,[11.43,13.91] (6.25)
	Real- Carry Forward	
Encoder/Decoder	20.36,[19.46,21.30] (0.08)	14.67,[14.11,15.24] (0.24)
Proposed	**20.16,[19.28,21.06] (0.28)**	**14.64,[14.09,15.20] (0.27)**
	Real- NO Carry Forward	
Encoder/Decoder	20.64,[19.74,21.56] (-0.2)	14.98,[14.43,15.56] (-0.08)
Proposed	20.41,[19.53,21.35] (0.03)	14.85,[14.28,15.41] (0.05)