LSTMs and Neural Attention Models for Blood Glucose Prediction: Comparative Experiments on Real and Synthetic Data

Sadegh Mirshekarian[1], Hui Shen[1], Razvan Bunescu[1,2], and Cindy Marling[1,2]

Abstract— **We have shown in previous work that LSTM networks are effective at predicting blood glucose levels in patients with type I diabetes, outperforming human experts and an SVR model trained with features computed by manually engineered physiological models. In this paper we present the results of a much larger set of experiments on real and synthetic datasets in what-if, agnostic, and inertial scenarios. Experiments on a more recent real-patient dataset, which we are releasing to the research community, demonstrate that LSTMs are robust to noise and can easily incorporate additional features, such as skin temperature, heart rate and skin conductance, without any change in the architecture. A neural attention module that we designed specifically for time series prediction improves prediction performance on synthetic data; however, the improvements do not transfer to real data. Conversely, using time of day as an additional input feature consistently improves the LSTM performance on real data but not on synthetic data. These and other differences show that behavior on synthetic data cannot be assumed to always transfer to real data, highlighting the importance of evaluating physiological models on data from real patients.**

I. INTRODUCTION AND MOTIVATION

Type 1 diabetes (T1D) is an autoimmune disease in which the pancreas fails to produce insulin, an essential hormone needed to control blood glucose levels. T1D is treated through exogenous supplies of insulin, either via multiple daily injections or by means of an insulin pump. To achieve and maintain good control, the person with T1D self-monitors blood glucose levels throughout the day, by testing blood from finger pricks and, sometimes, by using a continuous glucose monitoring (CGM) system with subcutaneous glucose sensors. When the blood glucose level is too high (hyperglycemia) or too low (hypoglycemia), the individual *reacts* to bring it back into range. Accurate forecasting of blood glucose levels would enable people with T1D to *proactively* intervene to prevent these conditions from occuring, thus enhancing health, safety, and quality of life. The value of modeling blood glucose levels has long been recognized, with attempts dating back to the 1960s [1]. A comprehensive review of BGL prediction strategies is available in [2].

The broad objective of the work described in this paper is to investigate and improve blood glucose level prediction using recurrent neural networks, using both simulated patient data and data collected from people with T1D on insulin pump therapy with CGM. It is a continuation of the machine learning research at Ohio University's SmartHealth Lab [3]–[8], where intelligent systems for diabetes management have been the focus of research for more than a decade. The rest of this paper is organized as follows: Section II describes the LSTM models and the new neural attention architecture developed for time series prediction. Sections III, IV and V explain our evaluation scenarios, the datasets we have used, and the experimental configurations for both standard and memory-augmented LSTM architectures. This is followed by results and discussion, categorized by dataset, in Section VI.

II. LSTMs for Time Series Prediction

The blood glucose level (BGL) prediction problem can be formally defined as predicting a *target* blood glucose value $\mathrm{BG}_{T+\tau}$ from a time series of blood glucose (BG) data given by $\langle \mathrm{BG}_t, \mathbf{e}_t \rangle,\ t = 1, 2, ..., T$, where BG_t is the blood glucose level at time t and $\mathbf{e}_t$ is the set of exogenous *events* such as meal and insulin. The *prediction horizon* or *time horizon* τ is relative to the present time T and is set to 30 or 60 minutes in the experiments reported in this paper. We sometimes use these terms to also refer to the absolute time $T + \tau$.

Recurrent neural networks (RNNs) are a type of neural network suitable for sequential data such as time series. Although very powerful in theory, vanilla RNNs are plagued by issues such as the vanishing gradient problem, which makes it harder for them to learn to carry information over long sequences [9]–[11]. The more advanced long-short term memory (LSTM) architecture allows RNNs to circumvent these issues and become much more competitive [12]. An LSTM node has two types of states that are passed along from the current to the next time step. The cell state $\mathbf{c}_t$, which holds/discards information according to the cell's mechanism, and the hidden state $\mathbf{h}_t$, used to compute the output $\mathbf{y}_t$. Each LSTM node also has *forget* (f), *input* (i) and *output* (o) gates which control how much the cell state value is preserved, updated with input values, and contributing to the hidden state, respectively. Each of the three gates has values in the range $[0, 1]$, with 0 and 1 representing closed and open gates. They are computed at each time step t using the following equations where σ is the sigmoid function:

$$\mathbf{i}_t = \sigma(W^{(i)}\mathbf{h}_{t-1} + U^{(i)}\mathbf{x}_t + \mathbf{b}^{(i)})$$
$$\mathbf{f}_t = \sigma(W^{(f)}\mathbf{h}_{t-1} + U^{(f)}\mathbf{x}_t + \mathbf{b}^{(f)})$$
$$\mathbf{o}_t = \sigma(W^{(o)}\mathbf{h}_{t-1} + U^{(o)}\mathbf{x}_t + \mathbf{b}^{(o)})$$

Once the gate values are computed, the new hidden state and

[1] School of EECS, Ohio University, Athens, Ohio, 45701, USA sm774113,hs138609,bunescu,marling@ohio.edu

[2] The Diabetes Institute, Heritage College of Osteopathic Medicine, Ohio University, Athens, Ohio 45701, USA

978-1-5386-1311-5/19/$31.00 ©2019 IEEE

cell state values are computed as follows:

$$\mathbf{z}_t = tanh(W^{(z)}\mathbf{h}_{t-1} + U^{(z)}\mathbf{x}_t + \mathbf{b}^{(z)})$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{z}_t$$

$$\mathbf{h}_t = \mathbf{o}_t \odot tanh(\mathbf{c}_t)$$

where the symbol $\odot$ stands for element-wise multiplication.

For the BGL prediction problem, the output corresponds to the BGL at the time horizon $T+\tau$, computed as a simple linear transformation of the hidden state at time T:

$$BG_{T+\tau} = \mathbf{v}^T\mathbf{h}_T + b$$

A. A New Memory-Augmented LSTM for Time Series

Traditionally, once trained, a parametric model does not have access to information about the training data beyond what it has learned and stored in its parameters. However, it has been shown that direct access to training data can lead to improved performance. Memory-augmented networks are a solution that has gained popularity recently [13], [14], in which the idea is to give the model access over a set of external memory slots, which can be written to and read from during training and testing.

In our study, when diabetes experts were asked to make BGL predictions for $T+\tau$ based on the patient history up to time T, they would often refer to *similar cases* in the past. The expectation was that similar BGL and activity histories prior to T will result in similar BGL behavior after T. To emulate this *case-based* prediction scenario, we introduce a general memory-augmented LSTM architecture (MemLSTM) specifically designed for time series prediction. The left of Figure 1 shows the 3 main modules of the MemLSTM architecture: the LSTM module, the memory module, and the feed-forward (FF) module.

The LSTM module is a conventional LSTM network, as explained in Section II. It scans the input sequence one element $\mathbf{x}_t$ at a time and recursively updates the hidden state vectors $\mathbf{h}_t$, up to the prediction time T. A weighted average $\bar{\mathbf{h}}_T = \sum_{t=T-\Delta}^{T} w_t\mathbf{h}_t$ of the last hidden state vectors back to $\mathbf{h}_{T-\Delta}$ is then computed to represent the current input, where $0 \leq \Delta \leq 60$ minutes is a hyper-parameter to be tuned for each dataset. Compared to using just the last hidden state, this trained weighted average was preferred due to slightly better results on development data.

The memory module stores the hidden state values $\mathbf{h}_t$ for each time step t up to 24 hours before T, together with the corresponding target value $BG_{t+\tau}$. Using $\bar{\mathbf{h}}_T$ as reference, attention weights a_i are computed for all examples in memory using a two-layer FF network which takes as input the concatenation of $\bar{\mathbf{h}}_T$ and $\mathbf{h}_i$ and outputs a real value representing the attention weight:

$$a_i = \tanh(W_{f2}(\tanh(W_{f1}[\bar{\mathbf{h}}_T; \mathbf{h}_i] + \mathbf{b}_{f1})) + b_{f2}) \qquad (1)$$

Note that this equation uses a weighted average of the last hidden state vectors $\bar{\mathbf{h}}_T$ as a representation for the current test example, as illustrated in Figure 1.

The FF module takes as input a vector $[\bar{\mathbf{h}}_T; a_*; \mathrm{BG}_{t_*+\tau}]$ that concatenates the averaged LSTM hidden state $\bar{\mathbf{h}}_T$ with the maximum attention weight a_* and the target value $\mathrm{BG}_{t_*+\tau}$ corresponding to the example t_* in memory with the maximum attention weight. This input is passed through one fully connected hidden layer followed by a single linear output neuron which generates the BGL prediction, as shown below:

$$\mathrm{BG}_{T+\tau} = [W_h, \mathbf{w}_a, \mathbf{w}_g][\bar{\mathbf{h}}_T; a_*; \mathrm{BG}_{t_*+\tau}] + b \qquad (2)$$

where $[W_h, \mathbf{w}_a, \mathbf{w}_g]$ are the corresponding parameters.

B. Comparison with Standard Neural Attention Models

The MemLSTM attention model differs significantly from the approach commonly used in the literature. Normally, the attention weights would be passed through a Softmax layer [13]. This has the effect of normalizing the attention values between 0 and 1 and, most importantly, the maximum value is pushed very close to 1, irrespective if how small it is in absolute terms. This makes sense for machine translation, where the correct target word always has a corresponding source word with a similar meaning. However, it does not make sense for case-based prediction of BG values, where there are many time steps for which the context and BG behavior is different from all past situations. It is therefore important not only to *not* normalize the attention weights, but also to provide the maximum attention weight a_* as input to the FF module that computes the BG prediction. We found that a_* varies significantly in magnitude from one test example to another, therefore the model needs to know its value in order to determine how much to count on the corresponding target BGL. In other words, if the closest example in memory $\mathbf{h}_{t_*}$ is still too different from the test example, the corresponding target value $\mathrm{BG}_{t_*+\tau}$ should be ignored. Furthermore, instead of using a weighted average over all the examples in memory, we used only the example with the maximum attention weight, which is less expensive computationally and leads to better results. The advantage of the new neural attention model was confirmed empirically: comparatively, using the traditional softmax-based attention model led to a substantial degradation in performance.

III. Evaluation Scenarios

Let E_τ be the set of events (e.g. meal, bolus, exercise) between the present prediction time T and the prediction horizon $T+\tau$, i.e. inside the *prediction range* $[T, T+\tau)$. Henceforth, we shall call E_τ the set of *what-if* events. We define three scenarios for the BG prediction problem, depending on how the system uses the events in E_τ.

A. The What-If scenario

If the model is given access to *what-if* events, we say that we have a What-If scenario. The purpose here is to have a model that can answer questions such as "What will my BGL be in 60 minutes if I eat a snack with 30 carbs 10 minutes from now". This would enable a system to use the BGL prediction model to recommend corrective actions at any time during the prediction range.

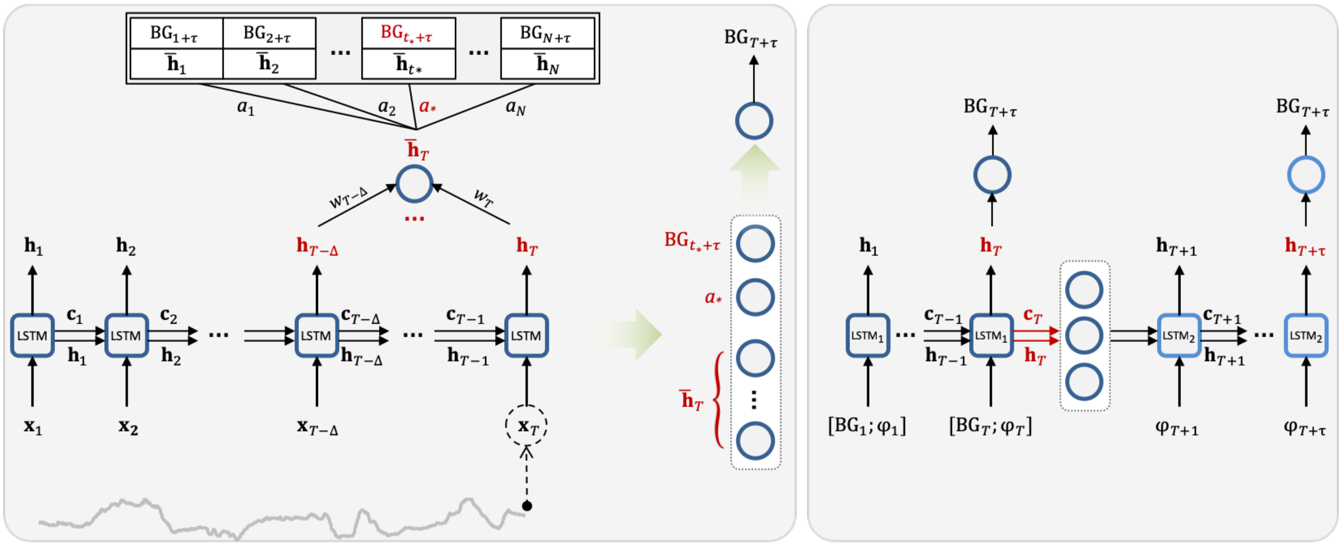

Fig. 1. [**Left**] The memory-augmented LSTM (MemLSTM) architecture is composed of 3 modules. The LSTM module (bottom) scans the input sequence of consecutive BGL readings, where $\mathbf{x}_t$ can be a vector containing meal, insulin and other activity information besides BGL. The memory module (top) is an array of past $\mathbf{h}_t$ values and their target $BG_{t+\tau}$. The feed-forward module (right) computes the BGL prediction $BG_{T+\tau}$ based on information provided by the LSTM and memory modules. a_* is the maximum attention weight matching the LSTM state with the memory content, and $BG_{t_*+\tau}$ is the value at the prediction horizon for the corresponding example at time t_*. The sample BG curve at the bottom is used only for illustration and is not necessarily synchronized in time with the inputs $\mathbf{x}_t$ above. [**Right**] The double-LSTM architecture used to capture information before and after prediction time T, to be used in the What-If scenario. Both LSTMs produce outputs that are used to compute their respective MSE losses, which are then added to form the overall model loss. The linear layers used to compute the two outputs do not share weights.

Because the BG levels are not available in the prediction range, exploiting the sequence of events in this range requires a second LSTM model, as shown in the architecture on the right in Figure 1. After the first LSTM scans the data up to time step T, the second LSTM scans the data, without its BG component, between T and $T + \tau$. The state of the second LSTM is initialized with the last state of the original LSTM, after being passed through a linear transformation followed by a non-linear $tanh$ activation. Note that to guide the first LSTM to learn useful representations, we computed the target using the last output of both LSTMs, instead of only the second one, and used a model loss equal to the sum of both LSTM losses with equal weight.

B. The Agnostic scenario

If the model is not given access to *what-if* events but is trained and tested on all examples, including those that do have *what-if* events, we say that we have an Agnostic scenario. To perform well, a model trained in this scenario would need to implicitly estimate life events that are likely to happen in the prediction range.

C. The Inertial scenario

If the model is trained and tested only on examples for which E_τ is empty, i.e., there are no life events in the prediction range, we say that we have an Inertial scenario. A model trained in this scenario predicts the BG level under the condition that the patient does not take any actions between now and the prediction horizon. The only difference from the Agnostic scenarios is that here the examples with *what-if* events are ignored during training and testing. Therefore, we expect the Inertial scenario to lead to more accurate predictions than the Agnostic scenario.

IV. Datasets

In this section, we describe the new real-patient dataset and two synthetic datasets that were used for experimental evaluation. The **OhioT1DM** dataset [15] contains data from six patients with type 1 diabetes who participated in an IRB-approved study for eight weeks each, between March 2016 and April 2017. Compared to the dataset used in [8], OhioT1DM has more accurate BGL measurements and more reliable information about meal and insulin events. The patients used the more accurate Medtronic Enlite® sensors for glucose monitoring, along with a Basis Peak fitness band, which provided additional physiological data such as heart rate (HR), skin temperature (ST), and skin conductance (SC). The carbs entered into the Bolus Wizard were also recorded for these patients. This dataset has now been released for research purposes.

Compared to real-patient data, time series data obtained from BGL simulators has a relatively well-defined, deterministic behavior, with a level of noise that can be controlled manually. We experimented with data from two diabetes simulators: AIDA and UVa/Padova. **AIDA** is an online BG simulator that is free for educational purposes. The physiological model used to compute BGL as a function of meal and insulin levels is described in [16], [17] and [18]. Overall, AIDA provides 40 simulated patient profiles that specify when a patient eats, sleeps, or injects insulin, and the profiles can be modified as necessary. The data obtained from the simulator consists of 24-hour-long sequences of meal, insulin and BGL, at a resolution of 15 minutes. Our experiments with the simulator showed that its BGL behavior is fairly predictable given meal and insulin information, so AIDA data can be a good start to make initial model adjustments. However, the utility of the dataset was limited by the lower time resolution and the 24-hour maximum sequence length.

UVa/Padova is a Type 1 Diabetes Metabolic Simulator (T1DMS), developed at the Universities of Virginia and Padova, henceforth referred to as UVa. It implements the dynamics of the human metabolic glucose-insulin system in the Matlab/Simulink environment [19]. Since 2008, UVa is the only FDA-approved simulator that can be used in lieu of pre-clinical animal testing for evaluation of certain diabetes treatment strategies. We used the newest distributed S2013 version [20], which has 30 patient profiles, evenly distributed in the adult, adolescent and child categories. We used the 10 adult profiles unmodified, but the provided interface was used to run the simulation on custom life habit scenarios. The following parameters were customized:

- Meal profiles: amount, timing, and duration of meals.
- Insulin treatment: amount and timing of basal and bolus.

The simulation was run for 90 days for each of the 10 patient profiles in 15 customized scenarios, amassing a total of 13,500 days of meal, insulin and BGL data. Out of these 10 profiles, the five hardest (obtaining the highest RMSE in preliminary evaluations) were selected for final experiments.

V. Experimental Setting

Obtaining good performance from RNNs and machine learning models in general requires careful tuning of their architectures and hyper-parameters. In this section, we describe the experimental settings used to evaluate our models on the real-patient and synthetic datasets.

A. Experimental configuration for AIDA

For AIDA, given that the example sequences have a length of 24 hours each and are independent of each other, we randomly partitioned the 600 days of data available for each patient into 100 days for testing, 100 days for development, and the remaining 400 days for training. For each run, training and testing was performed on each patient separately and the results were averaged over patients.

A single LSTM layer with 20 nodes was selected, with 12 hours used for backpropagation through time (BPTT). The BGL values are scaled by 1/600, while all other features are normalized to the range [0, 1/3]. The MSE objective is minimized using Adam [21] with a batch size of 512. We opted for a stepwise decreasing scheme of the learning rate in which 0.01 is chosen as the initial rate and 0.001 is chosen after meeting convergence criteria. Convergence was defined to be when performance on development data did not increase for 50 epochs, or a maximum allowable of 500 epochs was reached. Experiments with dropout are also reported, using variational dropout [22].

For MemLSTM experiments, the LSTM module was pre-trained separately as explained in Section II-A. During training with memory, a weighted average $\overline{\mathbf{h}}_T$ of the last 4 hidden state vectors (i.e. over the previous 60 minutes) is used to represent each example sequence in memory and during attention weight computation. The representations $\overline{\mathbf{h}}$ of all training examples were stored in memory, and updated after every epoch. The weights were initialized by sampling at random from the uniform distribution over [-0.5, 0.5].

B. Experimental configuration for UVa

UVa data exhibits a more realistic dynamic, and does not have the one-day sequence length limitation of AIDA. Out of the 90 consecutive days of data available for each patient, the 10 last days were selected as test, the 10 days before that as development, and the remaining 70 days as training data. BGL values were sampled every five minutes and all other variables were averaged inside the five minute intervals. Let ts_1, ts_2 and ts_3 be the time stamps associated with the beginning of training, development and test data, such that we have $ts_1 = 0$, $ts_2 = ts_1 + 70 * 24\text{hr}$ and $ts_3 = ts_2 + 10 * 24\text{hr}$. To avoid any overlaps between the three sets, the last prediction times for training and development datasets are limited to $T_e = ts_2 - \tau - 1$ and $ts_3 - \tau - 1$ respectively, so that $T_e + \tau$ does not go beyond the set's last time stamp. To avoid testing on points very close chronologically to the last development point and thus label-correlated to it, the first prediction time for the test datasets is limited to $T_b = ts_3 + 6\text{hr}$, i.e. 6 hours after the end of the development set. A similar limit is established for the first prediction time during tuning on development data. At each run, training and testing was performed on each patient separately and the results were averaged over patients.

A single LSTM layer with 20 nodes was used, with the BPTT sequence length shortened to 6 hours. For MemLSTM experiments, the weighted average representation $\overline{\mathbf{h}}_T$ was computed over the previous 60 minutes, corresponding to 12 state vectors sampled every 5 minutes. All other hyper-parameters were kept the same as for AIDA.

C. Experimental configuration for OhioT1DM

The UVa configuration was slightly modified for OhioT1DM. One major change was switching back to the RMSProp optimizer instead of Adam, which in our experiments achieved slightly better performance on the real-patient datasets. As for data, the same breakdown as the one for UVa was used: the last 10 days for testing, the previous 10 days for development, and the remaining days for training, while similarly filtering some of the early points to prevent overlaps. However, the number of examples available for OhioT1DM patients was close to half of the UVa data, so pre-training was deemed useful to alleviate overfitting. Toward this end, a bigger training and development dataset was created from the training and development sequences of all patients, respectively. The model was pre-trained with a two-step learning rate scheme that reduces the learning rate from 0.01 to 0.001 when performance on development data does not improve for 5 epochs, or a maximum of 100 epochs is reached, and stops according to the same criteria. The pre-trained model was then fine-trained on the training data of the actual test patient with a similar learning rate scheme, but with step values of 0.001 and 0.0001. Missing BGL values were linearly interpolated. However, any example in the dataset for which the target value $\mathrm{BG}_{T+\tau}$ is an interpolated BGL was discarded. For a contiguous sequence of missing values ending at T, an alternative imputation approach is to extrapolate, using only data before the missing sequence. The experimental results reported in Table VII do not show a consistent difference between the two approaches.

In the What-If scenario, the LSTMs have 1 layer of 20 nodes and the two MSE losses are summed together.

VI. Results and Discussion

For empirical comparisons, we used two baselines: $\mathbf{t_0}$ and **ARIMA**. The $\mathbf{t_0}$ baseline assumes that the target variable does not change, i.e. it predicts that $BG_{T+\tau} = BG_T$. Trained models are expected to be at least as good as this simple baseline. **ARIMA** stands for AutoRegressive Integrated Moving Average, a popular time series prediction approach. We use the `forecast` package in R which allows for automatic tuning of the ARIMA hyper-parameters p, q and r. A sequence length of between 4-7 days was shown to be the best for all of our experiments.

Experimental results on synthetic data are shown in Section VI-A), followed by results on real data in Section VI-B.

A. LSTM on synthetic data: The effect of memory, dropout, and time of day

We ran experiments on the two synthetic datasets in order to study the effect of dropout and additional features, as well as the utility of the memory-augmented network MemLSTM. To train ARIMA on the AIDA data where days of BG values were generated independently, we created one long sequence by chaining all days in random order and inserting a day of missing values between any two consecutive days of data. This enabled the use of the ARIMA implementation from the `forecast` package in R. The hyper-parameters $p = 6$ (order), $d = 1$ (degree of differencing), and $q = 1$ (order of the moving-average model) were tuned on the development data using grid search.

Tables I and II summarize the results on AIDA and UVa data respectively, both in the Inertial scenario. We use the acronyms BG for blood glucose, I for insulin delivered through bolus and basal, M for meals, and ToD for time of day. The following observations can be made from the results:

- When only blood glucose is used, the LSTM model achieves significantly better results than ARIMA. The gap is much bigger than what is observed on real patient data, an indication that the LSTM can effectively use information in the 6 hours of patient history.
- MemLSTM obtains better RMSE than LSTM. The improvements are statistically significant: a one-tailed paired t-test results in a p-value < 0.002.
- Time of Day (ToD) as an extra feature does not help.
- Dropout does not help on either of the datasets. Rates lower than 10% were also shown to be not helpful. The negative effect of dropout could be explained by the reduced amount of noise and the more deterministic behavior in synthetic data compared to real-patient data.

TABLE I

Average and standard deviation of RMSE results over 12 runs, obtained on 10 AIDA patients in the Inertial scenario

		Inertial	
Model	Features	30 min	60 min
t_0	BG	16.33	39.82
ARIMA	BG	5.59	16.48
LSTM	BG	$4.22_{0.04}$	$12.47_{0.13}$
LSTM	BG, I, M	$1.26_{0.03}$	$2.30_{0.04}$
MemLSTM	BG, I, M	$\mathbf{1.23}_{0.03}$	$\mathbf{2.27}_{0.03}$
LSTM	BG, I, M, ToD	$1.32_{0.03}$	$2.33_{0.04}$
LSTM (d = 0.1)	BG, I, M, ToD	$3.39_{0.06}$	$6.85_{0.22}$

TABLE II

Average and standard deviation of RMSE results over 12 runs, obtained on 5 UVa patients in the Inertial scenario

		Inertial	
Model	Features	30 min	60 min
t_0	BG	17.02	23.49
ARIMA	BG	12.00	18.66
LSTM	BG	$9.88_{0.40}$	$13.71_{0.27}$
LSTM	BG, I, M	$3.00_{0.11}$	$5.08_{0.31}$
MemLSTM	BG, I, M	$\mathbf{2.93}_{0.11}$	$\mathbf{4.92}_{0.32}$
LSTM	BG, I, M, ToD	$3.18_{0.66}$	$5.27_{0.29}$
LSTM (d = 0.1)	BG, I, M, ToD	$4.20_{0.14}$	$6.63_{0.28}$

Table III shows the number of predictions falling in the 5 areas from A to E in the Clarke Error Grid Analysis (CEGA) [23], a standard for evaluating the accuracy of BG sensors and BG predictions. Overall, the results show that the LSTM model makes fewer predictions in the costlier regions B, C, D, and E, when compared with the ARIMA baseline.

TABLE III

CEGA results for UVa, Inertial scenario, 60 min.

	A	B	C	D	E
LSTM	11,307	175	4	57	0
ARIMA	10,201	1,125	9	207	1

B. LSTM on real patient data with physiological sensor data

Table IV summarizes the first set of results for the OhioT1DM dataset. The following observations can be made from these results:

- Similar to synthetic data, the LSTM is able to use meal (M) and insulin (I) information more effectively than ARIMA.
- Contrary to synthetic data, dropout is clearly useful on this dataset.
- Unlike on synthetic data, using memory does not help performance on real data.

TABLE IV

Average and standard deviation of RMSE results over 20 runs on the OhioT1DM dataset, for LSTM with and without variational dropout

		Agnostic		Inertial	
Model	Features	30 min	60 min	30 min	60 min
t_0	BG	22.60	36.66	21.67	34.43
ARIMA	BG	20.17	33.47	19.36	31.45
LSTM (d = 0.0)	BG	$19.51_{0.17}$	$32.04_{0.28}$	$18.64_{0.10}$	$29.64_{0.23}$
LSTM (d = 0.1)	BG	$19.07_{0.12}$	$31.11_{0.16}$	$18.72_{0.13}$	$29.52_{0.15}$
LSTM+Mem (d = 0.1)	BG	$19.09_{0.11}$	$31.09_{0.16}$	$18.75_{0.16}$	$29.55_{0.15}$
LSTM (d = 0.0)	BG, I, M	$19.01_{0.19}$	$30.94_{0.82}$	$18.35_{0.24}$	$29.42_{0.54}$
LSTM (d = 0.1)	BG, I, M	$\mathbf{18.74}_{0.17}$	$\mathbf{30.63}_{0.27}$	$\mathbf{18.07}_{0.10}$	$28.32_{0.21}$
LSTM+Mem (d = 0.1)	BG, I, M	$18.77_{0.17}$	$30.65_{0.27}$	$18.09_{0.10}$	$\mathbf{28.28}_{0.28}$

TABLE V

Average and standard deviation of RMSE results over 20 runs on the OhioT1DM dataset, for LSTM with 10% variational dropout when different features are used

							Agnostic		Inertial	
BG	I	M	SC	HR	ST	ToD	30 min	60 min	30 min	60 min
●	○	○	○	○	○	○	$19.07_{0.12}$	$31.11_{0.16}$	$18.72_{0.13}$	$29.52_{0.15}$
●	○	○	○	○	○	●	$19.11_{0.09}$	$31.11_{0.18}$	$18.75_{0.18}$	$29.32_{0.30}$
●	●	●	○	○	○	○	$18.74_{0.17}$	$30.63_{0.27}$	$18.07_{0.10}$	$28.32_{0.21}$
●	●	●	○	○	○	●	$18.80_{0.16}$	$30.56_{0.23}$	$18.13_{0.13}$	$28.26_{0.22}$
●	●	●	●	○	○	○	$18.81_{0.21}$	$30.31_{0.19}$	$18.19_{0.11}$	$28.29_{0.25}$
●	●	●	●	○	○	●	$18.81_{0.16}$	$30.18_{0.20}$	$18.10_{0.09}$	$\mathbf{28.19}_{0.15}$
●	●	●	●	●	○	○	$18.77_{0.16}$	$30.28_{0.19}$	$18.10_{0.14}$	$28.30_{0.32}$
●	●	●	●	●	○	●	$\mathbf{18.70}_{0.13}$	$\mathbf{30.17}_{0.22}$	$18.07_{0.08}$	$28.20_{0.19}$
●	●	●	●	●	●	○	$18.76_{0.17}$	$30.40_{0.23}$	$18.10_{0.08}$	$28.37_{0.29}$
●	●	●	●	●	●	●	$18.83_{0.17}$	$30.43_{0.23}$	$\mathbf{17.99}_{0.10}$	$28.20_{0.19}$

After establishing that dropout is useful on OhioT1DM data, we conducted a more thorough set of experiments to see the effect of heart rate, skin temperature and skin conductance on the LSTM performance. The following observations can be made from the results shown in Table V:

- The biggest improvement is obtained when adding meal (M) and insulin (I) data. This was also evident from Table IV.
- A one-tailed paired t-test over the 10 pairs of results from the Inertial scenario with and without time of day showed that the improvement from adding time of day as a real-valued feature is statistically significant. This was not the case for Agnostic scenario results. However, when combined to form 20 pairs, the improvement was statistically significant.
- When skin conductance (SC) and heart rate (HR) are added to blood glucose (BG), insulin (I), meals (M) and time of day (ToD), they improve the results in both scenarios, for both 30 and 60 minutes.

Lastly, we ran evaluations in the What-If scenario. As shown in Table VI, using meal and insulin events between T and $T+\tau$ clearly improves results over the corresponding feature sets from Table V, especially for the prediction horizon of 60 minutes. We also ran the same experiments, using linear extrapolation to fill any contiguous sequence of missing BG values ending at T. The results in Table VII show that no consistent difference can be observed between the two imputation approaches.

TABLE VI

Average and standard deviation of RMSE results over 20 runs on the OhioT1DM dataset, in the What-if scenario

		What-if	
Model	Features	30 min	60 min
LSTM (d = 0.1)	BG, I, M	$18.19_{0.18}$	$29.12_{0.21}$
	BG, I, M, ToD	$18.18_{0.21}$	$29.07_{0.29}$
	BG, I, M, SC, HR, ST, ToD	$\mathbf{18.10}_{0.10}$	$\mathbf{29.04}_{0.26}$

TABLE VII

RMSE results over 20 runs on the OhioT1DM dataset, in the What-if scenario, using linear extrapolation to impute sequences of missing values ending at T

		What-if	
Model	Features	30 min	60 min
LSTM (d = 0.1)	BG, I, M	$18.19_{0.15}$	$29.27_{0.22}$
	BG, I, M, ToD	$\mathbf{18.11}_{0.18}$	$29.08_{0.28}$
	BG, I, M, SC, HR, ST, ToD	$18.15_{0.12}$	$\mathbf{28.93}_{0.23}$

Table VIII shows the number of predictions falling in the 5 areas from A to E in the Clarke Error Grid Analysis (CEGA). Overall, the results show that the LSTM model makes fewer predictions in the costlier regions B, C, D, and E, when compared with the ARIMA baseline.

TABLE VIII

CEGA RESULTS FOR OHIOT1DM, WHAT-IF SCENARIO, 60 MIN.

	A	B	C	D	E
LSTM	12,627	2,733	0	115	0
ARIMA	11,236	3,823	60	353	3

C. Comparison with Other Approaches on Real Data

The LSTM results presented in this paper were first described in an MS thesis that was defended by the first author in Spring 2018. Following that, we made the OhioT1DM Dataset publicly available for research purposes via a data use agreement and held the first Blood Glucose Level Prediction (BGLP) Challenge [24] at IJCAI in July, 2018. The goal was to bring researchers together to compare the efficacy of different prediction approaches on a standard set of real patient data. Compared to the 7 teams who participated, the What-If results reported in Table VII for the 30 minute prediction horizon using the double LSTM architecture (RMSE between 18.11 and 18.19) rank us first among results submitted by the original deadline (June 7). The overall ranking however needs to be considered with caution because the participating systems were not necessarily evaluated using exactly the same experimental setting. To enable reproducibility and future experimental comparions, we are making our code publicly available on the SmartHealth Lab web site at `http://smarthealth.cs.ohio.edu/nih.html`.

ACKNOWLEDGMENT

This work was supported by grant 1R21EB022356 from the National Institutes of Health (NIH). The authors gratefully acknowledge the contributions of Emeritus Professor of Endocrinology Frank Schwartz, MD. We would also like to thank the anonymous people with type 1 diabetes who shared their blood glucose, insulin, and life event data. The experimental procedures involving human subjects described in this paper were approved by the Institutional Review Board.

REFERENCES

[1] A. Boutayeb and A. Chetouani, "A critical review of mathematical models and data used in diabetology," *Biomedical Engineering Online*, vol. 5, no. 43, 2006.

[2] S. Oviedo, J. Vehí, R. Calm, and J. Armengol, "A review of personalized blood glucose prediction strategies for T1DM patients," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 33, no. 6, p. e2833, 2017.

[3] M. Wiley, R. Bunescu, C. Marling, J. Shubrook, and F. Schwartz, "Automatic detection of excessive glycemic variability for diabetes management," in *Proceedings of the 10th International Conference on Machine Learning and Applications*. Honolulu, Hawaii: IEEE Computer Society, 2011, pp. 1–7.

[4] C. Marling, M. Wiley, R. C. Bunescu, J. Shubrook, and F. Schwartz, "Emerging applications for intelligent diabetes management," *AI Magazine*, vol. 33, no. 2, pp. 67–78, 2012.

[5] R. Bunescu, N. Struble, C. Marling, J. Shubrook, and F. Schwartz, "Blood glucose level prediction using physiological models and support vector regression," in *Proceedings of the IEEE 12th International Conference on Machine Learning and Applications (ICMLA)*. Miami, FL: IEEE, December 2013, pp. 135–140.

[6] K. Plis, R. Bunescu, C. Marling, J. Shubrook, and F. Schwartz, "A machine learning approach to predicting blood glucose levels for diabetes management," in *Proceedings of the AAAI Workshop on Modern Artificial Intelligence for Health Analytics (MAIHA)*. Quebec City, Canada: AAAI Press, July 2014.

[7] C. Marling, L. Xia, R. Bunescu, and F. Schwartz, "Machine learning experiments with noninvasive sensors for hypoglycemia detection," in *Proceedings of IJCAI 2016 Workshop on Knowledge Discovery in Healthcare Data*, New York, NY, July 2016, pp. 1–6.

[8] S. Mirshekarian, R. Bunescu, C. Marling, and F. Schwartz, "Using LSTMs to learn physiological models of blood glucose behavior," in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, 2017, pp. 2887–2891.

[9] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen netzen," Ph.D. dissertation, Institut fur Informatik, Technische Universitat Munich, 1991.

[10] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, March 1994.

[11] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Networks*. IEEE Press, 2001.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, November 1997.

[13] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2014.

[14] J. Weston, S. Chopra, and A. Bordes, "Memory networks," *International Conference on Learning Representations*, pp. 1–14, 2015.

[15] C. Marling and R. Bunescu, "The OhioT1DM dataset for blood glucose level prediction," in *Proceedings of the 3^{rd} International Workshop on Knowledge Discovery in Healthcare Data*, 2018. [Online]. Available: http://smarthealth.cs.ohio.edu/OhioT1DM-dataset.html

[16] E. Lehmann and T. Deutsch, "A physiological model of glucose-insulin interaction in type 1 diabetes mellitus," *Journal of Biomedical Engineering*, vol. 14, no. 3, pp. 235–242, 1992.

[17] J. R. Guyton, R. O. Foster, J. S. Soeldner, M. H. Tan, C. B. Kahn, L. Koncz, and R. E. Gleason, "A model of glucose-insulin homeostasis in man that incorporates the heterogeneous fast pool theory of pancreatic insulin release," *Diabetes*, vol. 27, no. 10, pp. 1027–1042, 1978.

[18] M. Berger and D. Rodbard, "Computer simulation of plasma insulin and glucose dynamics after subcutaneous insulin injection," *Diabetes Care*, vol. 12, no. 10, pp. 725–736, 1989.

[19] B. P. Kovatchev, M. Breton, C. Dalla Man, and C. Cobelli, "In silico preclinical trials: A proof of concept in closed-loop control of type 1 diabetes," *Journal of Diabetes Science and Technology*, vol. 3, no. 1, pp. 44–55, 2009.

[20] C. Dalla Man, F. Micheletto, D. Lv, M. Breton, B. Kovatchev, and C. Cobelli, "The UVA/PADOVA type 1 diabetes simulator: New features," *Journal of Diabetes Science and Technology*, vol. 8, no. 1, pp. 26–34, 2014.

[21] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations 2015*, pp. 1–15, 2015.

[22] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Conference on Neural Information Processing Systems (NIPS)*, 2016.

[23] B. P. Kovatchev, L. A. Gonder-Frederick, D. J. Cox, and W. L. Clarke, "Evaluating the accuracy of continuous glucose-monitoring sensors: Continuous glucose-error grid analysis illustrated by TheraSense Freestyle Navigator data," *Diabetes Care*, vol. 27, no. 8, pp. 1922–1928, 2004.

[24] K. Bach, R. Bunescu, O. Farri, A. Guo, S. Hasan, Z. Ibrahim, C. Marling, J. Raffa, J. Rubin, and H. Wu, Eds., *Proceedings of the 3^{rd} International Workshop on Knowledge Discovery in Healthcare Data*, ser. CEUR Workshop Proceedings, no. 2148, 2018. [Online]. Available: http://ceur-ws.org/Vol-2148