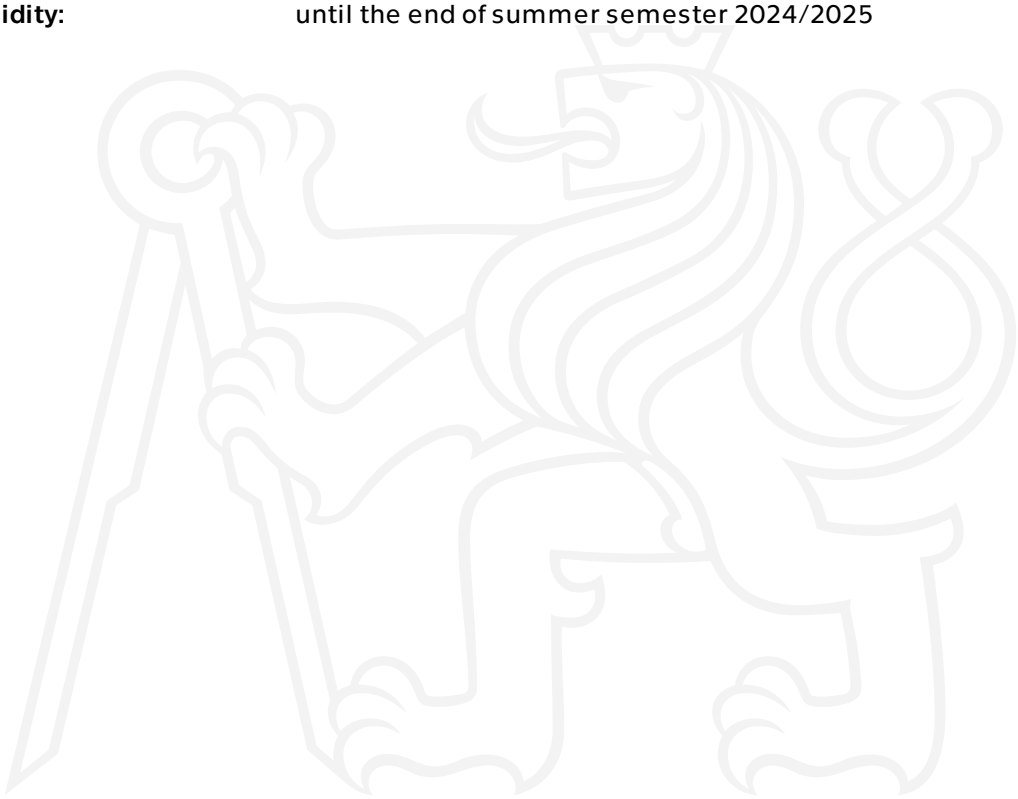
FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

# Assignment of master's thesis

| | |
|---|---|
| **Title:** | Improving blood glucose level prediction models |
| **Student:** | Bc. Ladislav Floriš |
| **Supervisor:** | Ing. Daniel Vašata, Ph.D. |
| **Study program:** | Informatics |
| **Branch / specialization:** | Knowledge Engineering |
| **Department:** | Department of Applied Mathematics |
| **Validity:** | until the end of summer semester 2024/2025 |



*Electronically approved by Ing. Magda Friedjungová, Ph.D. on 14 February 2024 in Prague.*

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

## Instructions

Type 1 diabetes disrupts normal blood glucose regulation due to the destruction of insulin-producing cells, necessitating insulin therapy through injections or insulin pumps. Consumer devices can forecast blood glucose levels by leveraging data from blood glucose sensors and other sources. Such predictions are valuable for informing patients
about their blood glucose trajectory and supporting various downstream applications. Numerous machine-learning models have been explored for blood glucose prediction.

The goal of the thesis is to research and improve machine learning models for blood glucose level prediction. One of the possible directions is to focus on Legendre memory units instead of common LSTM units or to use transformers (e.g., autoformer or informer). Moreover, for all those models, focusing on non-standard optimizers like the Lion optimizer might be beneficial.

Details assignment points:
1) Research the current state-of-the-art approaches for blood glucose predictions based on multivariate inputs.
2) Investigate promising models like LMU or transformer architectures like autoformer or informer that have the potential to improve performance. Select at least two of the most promising approaches.
3) Use the Ohio dataset to research the previously selected models experimentally. Experiment with various hyperparameters and training strategies.
4) Evaluate and discuss the results. Focus on the comparison with existing studies.

*Electronically approved by Ing. Magda Friedjungová, Ph.D. on 14 February 2024 in Prague.*

Master's thesis

# IMPROVING BLOOD GLUCOSE LEVEL PREDICTION MODELS

**Bc. Ladislav Floriš**

Faculty of Information Technology
Department of Applied Mathematics
Supervisor: Ing. Daniel Vašata, Ph.D.
May 9, 2024

Czech Technical University in Prague
Faculty of Information Technology


Citation of this thesis: Floriš Ladislav. *Improving blood glucose level prediction models.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2024.

# Contents

# List of Figures

# List of Tables

*I would like to thank my supervisor, Ing. Daniel Vašata, Ph.D., for his support and guidance throughout the writing of this thesis. His insightful contributions were crucial to the advancements presented in this work on blood glucose level prediction.*
*I also wish to express my appreciation to all who have supported me, contributing to the successful completion of my studies at CTU.*

# Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended. In accordance with Section 2373(2) of Act No. 89/2012 Coll., the Civil Code, as amended, I hereby grant a non-exclusive authorization (licence) to utilize this thesis, including all computer programs that are part of it or attached to it and all documentation thereof (hereinafter collectively referred to as the "Work"), to any and all persons who wish to use the Work. Such persons are entitled to use the Work in any manner that does not diminish the value of the Work and for any purpose (including use for profit). This authorisation is unlimited in time, territory and quantity.

In Prague on May 9, 2024

# Abstract

This work addresses the task of predicting blood glucose levels in patients with type 1 diabetes. Models based on Transformer architecture and Legendre Memory Units (LMU) were explored. The application of LMUs in this work represents their first use for blood glucose level prediction. Employing multivariate time series, predictions are made with 30-minute and 60-minute horizons. Models were trained and evaluated using the OhioT1DM dataset, which includes eight weeks of data from 12 distinct patients. The dataset consists of two editions, released in 2018 and 2020.

Performance was measured using Root Mean Square Error (RMSE), and Clarke Error Grid Analysis was utilized to evaluate clinical accuracy. LMUs achieved an RMSE of 18.17 mg/dl for the 30-minute horizon and 30.33 mg/dl for the 60-minute horizon, in the 2018 edition. In the 2020 edition, the RMSEs were 18.56 mg/dl and 32.57 mg/dl for the 30-minute and 60-minute horizons, respectively.

LMUs were proven to match and, in smaller datasets (2018 edition of OhioT1DM), even outperform the state-of-the-art models.



**Keywords** type 1 diabetes, blood glucose prediction, time series forecasting

# Abstrakt

Tato práce adresuje problém predikce hladiny glukózy v krvi u pacientů s diabetem 1. typu. Pro účel predikce byli aplikovány modely založené na Transformer architektuře a Legendre Memory Units (LMU). Aplikace LMU je v této práci taky první použití těchto modelů pro účel predikce hladiny glukózy v krvi. Modely predikovali budoucí hodnoty hladiny glukózy na základě vícerozměrných časových řad na vstupu a byli experimentálně hodnoceny na 30minutovém a 60minutovém predikčním horizontu. Modely byly trénovány a hodnoceny na datasetu OhioT1DM, který obsahuje osm týdnů dat od 12 různých pacientů. Dataset se skládá ze 2 edic, které byly vydány v letech 2018 a 2020.

Přesnost modelů byla hodnocena pomocí Root Mean Square Error (RMSE) a k vyhodnocení klinické přesnosti byla použita Clarke error grid analýza. LMU dosáhly RMSE 18.17 mg/dl pro 30minutový horizont a 30.33 mg/dl pro 60minutový horizont v edici OhioT1DM z roku 2018. V edici z roku 2020 byly RMSE 18.56 mg/dl a 32.57 mg/dl.

Bylo prokázáno, že LMU dosahují, a na menších datasetech (edice OhioT1DM 2018), dokáží i překonat stávající state-of-the-art modely.

**Klíčová slova** diabetes typu 1, predikce hladiny krevní glukózy, prognóza časových řad

# List of abbreviations

| | |
|---|---|
| T1D | Type 1 diabetes |
| BG | Blood glucose |
| FBG | Fingerstick blood glucose |
| CGM | Continuous glucose monitoring |
| PH | Prediction horizon |
| ML | Machine learning |
| RNN | Recurrent neural network |
| LSTM | Long short-term memory |
| LMU | Legendre memory unit |
| GRU | Gated recurrent unit |
| RMSE | Root mean square error |
| MSE | Mean square error |
| CEGA | Clarke error grid analysis |
| IST | Insulin in the subcutaneous tissue |
| CG | Carbohydrates in the gut |

# Introduction

Type 1 diabetes (T1D) is an autoimmune disorder causing abnormal blood glucose (BG) levels due to the body's incapability to produce insulin. Treatment primarily involves the patient's active involvement in managing their condition, which includes regularly monitoring glucose levels, administering insulin, and controlling diet and exercise. Although T1D cannot be cured, effective management can prevent complications by maintaining BG levels within the recommended range. Recent developments in diabetes treatment technologies and wearable devices have made collecting and processing relevant patient data easier.

Accurate prediction of BG levels in patients with T1D is crucial for developing tools that assist in making informed treatment decisions and for systems that automate insulin delivery. Such systems include blood glucose alarms that alert patients about upcoming hyperglycemic or hypoglycemic events and closed-loop systems used for automatic insulin delivery. These closed-loop systems utilize sensor-based glucose readings and potentially other data sources to compute insulin dosages, which are forwarded to an insulin pump to deliver the insulin.

Many machine learning (ML) based models have been proposed for the task of BG prediction, see Chapter 2 for detail. Some of the explored models were ARIMA, support-vector-machines (SVM), and neural networks. A popular choice of neural networks are recurrent neural networks (RNN), particularly LSTM.

## Objectives

This thesis aims to research and improve upon the current state-of-the-art methods for predicting blood glucose levels using multivariate inputs. This involves investigating promising model architectures, as well as training and data pre-processing techniques, with the aim of enhancing the performance of proposed ML models.

The training and evaluation of these models are conducted using the standard OhioT1DM dataset [1]. The best-performing models are compared against existing state-of-the-art approaches, and their clinical accuracy is evaluated. The prediction horizons are 30 and 60 minutes.

## Overview

The thesis is organized into the following chapters:

- Chapter 1 introduces the condition of T1D, the theory behind BG predictions, and datasets used to develop models for BG prediction, including the OhioT1DM dataset.
- Chapter 2 reviews related literature and studies.
- Chapter 3 explains the pre-processing applied to the dataset and the creation of new features.
- Chapter 4 puts forward the theory behind proposed ML models for BG prediction.
- Chapter 5 describes the model training and tuning and compares the proposed models and different training strategies.
- Chapter 6 showcases the best model, its performance, and clinical accuracy and compares it against the related studies.

# Chapter 1
# Background

## 1.1 Type 1 Diabetes

The introduction presented basic information about Type 1 Diabetes. This section provides a more detailed exploration of the condition, beginning with the challenges of BG regulation and dynamics.

The primary goal of T1D treatment is to maintain BG within a safe range, similar to that of a healthy individual, known as euglycemia. The typical euglycemic range cited in the research is approximately 60 to 140 mg/dl or 3.3 to 7.8 mmol/l. This range is commonly used for diabetes tests [2].

Hypoglycemia occurs when BG levels fall below 3.3 mmol/l or 60 mg/dl. Symptoms include anxiety, sweating, and hunger, progressing to more severe neurological effects such as behavioral changes, cognitive dysfunction, and, in extreme cases, seizures and coma. The lower the glucose level, the more severe the symptoms tend to be. Patients aim to avoid hypoglycemia due to its potential to severely impair functionality or be life-threatening [3].

Hyperglycemia is defined by BG levels exceeding 180 mg/dl or 10 mmol/l. Chronic levels of BG above this threshold can produce noticeable organ damage over time. Symptoms of hyperglycemia are usually benign, like dry mouth and polyuria, but can be more severe if hyperglycemia develops into ketoacidosis. The greatest danger of chronic hyperglycemia is the long-term effects, especially on the microvascular system. These effects may be life-threatening and include damage to the eye, kidneys, nerves, heart, and the peripheral vascular system [4].

### 1.1.1 Blood Glucose Dynamics

When analyzing BG dynamics, factors impact BG levels by either increasing or decreasing the BG.

The primary factor that causes BG levels to rise is the consumption of carbohydrates. These are metabolized into glucose, a simple sugar that is a

key energy source for various bodily functions. The time it takes for glucose to become available in the bloodstream after eating can vary depending on the food consumed. The duration and rate at which glucose is released into the blood are measured by the Glycemic Index, which ranks foods accordingly [5]. Additionally, the rate of glucose release into the bloodstream is specific to each individual and can vary according to different life situations.

Additional factors that can increase BG levels include stress, anaerobic exercise, and the action of glucagon — a hormone that prompts the release of stored glucose from the liver, among other influences.

The primary factor that reduces BG levels is insulin. There are various types of insulin that differ in their onset of action and overall duration of their effect, leading to differing ways of acting on the BG levels. Insulin is typically administered through insulin pens or insulin pump injections. Insulin pumps deliver insulin primarily in two ways: basal and bolus. Basal insulin involves regular, small infusions of short-acting insulin, while bolus doses are larger, single-time injections, typically before meals or as corrective doses to manage BG levels. As was demonstrated by Davidson et al. [6], the effect of insulin on BG levels also varies from person to person, depending on individual insulin sensitivity. This sensitivity is influenced by factors such as age, weight, and the average amount of insulin administered in recent weeks.

Additional factors that can lower BG include aerobic exercise and alcohol consumption.

## 1.2 Blood Glucose Prediction

Martin H. Kroll [7] demonstrated that BG variations include a deterministic component and that it can be described as a nonlinear oscillatory or chaotic system that can be modeled.

The BG prediction task can be defined as univariate time-series forecasting if only past BG levels are used for predictions or multivariate time-series forecasting if multiple features, including past BG levels, are used to make predictions. This work focuses on the multivariate version. Features influencing BG levels include insulin, carbohydrate consumption, physical activity, stress, illness, and others.

Formally, as the input, the model takes $n$ successive previous measurements of feature vectors in past $n$ time steps $t_1, t_2, \ldots, t_n$ from interval $[t-(n-1)\Delta t, t]$, where $t_i = t - \Delta t(n-i)$, as an input. Hence, extending $n$ or $\Delta t$ brings a longer historical context to the model. Each component of a feature vector $\mathbf{x_s} = (x_{s;1}, x_{s;2}, \ldots, x_{s;p})$ is associated with one of the $p$ features. The output is the prediction $\hat{G}_{t+\mathrm{PH}}$ of the BG level $G_{t+\mathrm{PH}}$ at time $t+\mathrm{PH}$, where PH is the prediction horizon indicating how long into the future the predictions are being made. In this work, the model, represented by a function $f_\Delta(\mathbf{x_{t_1}}, \ldots, \mathbf{x_{t_n}})$, analogously to [8] predicts the change $G_{t+\mathrm{PH}} - G_t$.

It is expected that with a longer PH, the prediction performance will decrease. BG takes time to change, and as PH increases, the BG has further opportunities to develop, leading to a wider range of possible BG values.

The aim is to find a model to predict the future BG values at a given PH with the smallest error and generalize well for various BG trajectories.

## 1.3 Overview of T1D Datasets

Multiple datasets are utilized to develop BG prediction models. A frequently used dataset is the OhioT1DM dataset [1], accessible to researchers through a formal access request. Additional notable datasets include the Tidepool dataset [9], OpenAPS Data Commons [10], and DiaTrend [11]. The mentioned datasets can be accessed upon access request, which can be made either directly to the authors or via the platforms hosting them.

To the best of the authors' knowledge, no datasets are immediately available for free download without prior inquiry, except for the Type 1 Diabetes Blood Glucose Prediction dataset [12]. This dataset is hosted on the Kaggle platform and was previously published by the author of this thesis. It contains five months of data from a single patient and is freely downloadable.

Additionally, many studies utilize synthetically generated data for model evaluation, with the UVA/Padova simulator being a frequently used generator [13].

This thesis uses the OhioT1DM dataset[1] [1], frequently used in research, it provides a wealth of related studies for comparison. This extensive use allows for a comprehensive evaluation of the achieved results against a broad selection of existing works.

## 1.4 OhioT1DM Dataset

The OhioT1DM dataset is available in two editions, from 2018 and 2020, and the models developed in this thesis are evaluated on both versions. This dataset includes data collected over eight weeks from 12 individuals with T1D, divided equally between the two editions. Each participant used a smartphone application and a fitness band to log daily activities. Patients wore either Medtronic 530G or 630G insulin pumps and used Medtronic Enlite CGM sensors throughout the data collection period, see Figure 1.1. The insulin used by the patients was either Humalog or Novalog. The first cohort from 2018 of 6 individuals wore Basis Peak fitness bands. The second cohort from 2020 of 6 individuals wore the Empatica Embrace. The full list of all features available is as follows [1]:

[1] Available at `http://smarthealth.cs.ohio.edu/OhioT1DM-dataset.html`.


**(a)** Medtronic 530G insulin pump [14]

**(b)** Medtronic Enlite CGM sensor [15]

**Figure 1.1** T1D equipment used by patients from the OhioT1DM dataset.

1. <patient> The patient ID number and insulin type.
2. <glucose level> CGM data, recorded every 5 minutes.
3. <finger stick> BG values obtained through self-monitoring by the patient.
4. <basal> The rate at which basal insulin is continuously infused.
5. <temp basal> A temporary basal insulin rate that supersedes the patient's normal basal rate.
6. <bolus> Insulin delivered to the patient. The most common bolus type, normal, delivers all insulin at once. Other bolus types can stretch out the insulin dose over the period between ts begin and ts end.
7. <meal> The self-reported time and type of a meal, plus the patient's carbohydrate estimate for the meal.
8. <sleep> The times of self-reported sleep, plus the patient's subjective assessment of sleep quality: 1 for Poor; 2 for Fair; 3 for Good.
9. <work> Self-reported times of going to and from work. Intensity is the patient's subjective assessment of physical exertion on a scale of 1 to 10, with 10 being the most physically active.
10. <stressors> Time of self-reported stress.
11. <hypo event> Time of self-reported hypoglycemic episode.
12. <illness> Time of self-reported illness.
13. <exercise> Time and duration, in minutes, of self-reported exercise. Intensity is the patient's subjective assessment of physical exertion on a scale of 1 to 10, with 10 being the most physically active.
14. <basis heart rate> Heart rate, aggregated every 5 minutes.

15. <basis gsr> Galvanic skin response. The data was aggregated every 5 minutes and 1 minute for the 2018 and 2020 cohorts, respectively.

16. <basis skin temperature> Skin temperature, in degrees Fahrenheit, aggregated every 5 minutes and 1 minute for the 2018 and 2020 cohorts, respectively.

17. <basis air temperature> Air temperature, in degrees Fahrenheit, aggregated every 5 minutes.

18. <basis steps> Step count, aggregated every 5 minutes.

19. <basis sleep> Times when the sensor band reported that the subject was asleep. For the 2018 cohort, there is also a numeric estimate of sleep quality.

20. <acceleration> Magnitude of acceleration, aggregated every 1 minute.

Features *basis heart rate*, *basis air temperature*, and *basis steps* are available for the 2018 cohort only, while feature *acceleration* is available for the 2020 cohort only.

There are two XML files for each data contributor: one for training and validation data and another for testing data. Thus, there are 24 XML files, with each of the 12 contributors having two files.

It is noted that for the purposes of this thesis, bolus insulin that is administered over a period of time was treated as if delivered all at once. Using the true stretched-out insulin slightly negatively affected the model performance. This can possibly be attributed to the fact that the models with the stretched-out insulin input had less information about the actual insulin dose to be delivered compared to the case when the whole insulin dose to be delivered was given to the model at once. However, this effect is limited because most doses are delivered at once.

### 1.4.1 CGM Sensor Accuracy

It is important to note that CGM sensors do not directly measure BG but rather glucose levels in the interstitial fluid. There is a lag between BG and interstitial glucose, which ranges from 5 to 10 minutes, according to Ananda Basu et al. [16]. Furthermore, errors also come from the sensor itself due to various factors.

The sensor lag is particularly significant in clinical settings, especially during rapid changes in BG levels. For example, during episodes where BG levels are quickly rising or falling, the glucose concentration can reach hypoglycemic or hyperglycemic range within a short period. Under such circumstances, patients might experience symptoms of hyperglycemia or hypoglycemia, even though the sensor indicates that glucose levels are within a safe range.

Due to potential errors and the time lag associated with CGM sensors, FBG samples, when available, can offer a more accurate reflection of current BG levels.

## 1.5 OhioT1DM Data Analysis

This section conducts an exploratory data analysis. A selection of features is chosen, which were further used in later chapters for model development.

Table 1.1 presents the number of training and testing BG samples available for each patient. The number of train and test samples is relatively consistent across patients. Patient 588 has the highest number of training samples, totaling 12640, while patient 552 has the fewest, with 9080. For testing samples, patient 540 has the maximum, with 2896 samples, and patient 552 has the minimum, at 2364.

**Table 1.1** Patient BG samples.

| Cohort | Patient ID | Train Samples | Test Samples |
|---|---|---|---|
| 2018 | 559 | 10796 | 2514 |
| 2018 | 563 | 12124 | 2570 |
| 2018 | 570 | 10982 | 2745 |
| 2018 | 575 | 11866 | 2590 |
| 2018 | 588 | 12640 | 2791 |
| 2018 | 591 | 10847 | 2760 |
| 2020 | 540 | 11947 | 2896 |
| 2020 | 544 | 10623 | 2716 |
| 2020 | 552 | 9080 | 2364 |
| 2020 | 567 | 10858 | 2389 |
| 2020 | 584 | 12150 | 2665 |
| 2020 | 596 | 10877 | 2743 |

Table 1.2 shows basic descriptive statistics for both OhioT1DM cohorts combined. They include the number of samples, mean, standard deviation, minimum and maximum, and 25, 50, and 75 percentile values. This presents an initial insight into the feature properties. It can be seen that the range of BG values is from 40 to 400, which also is the effective range of the Medtronic Enlite CGM sensor [17]. The average BG level is 159.58 mg/dl, with a similar average for FBG at 158.16 mg/dl. The typical bolus insulin dosage recorded is approximately 5.95 units, and the basal insulin rate averages about 1.01 units per hour.

To better understand the distribution of individual features, histograms of BG, FBG, carbohydrates, bolus, and basal insulin are presented in Figure 1.2 for glucose-related features and Figure 1.3 for insulin and carbohydrate features. The histograms for BG and FBG exhibit similar patterns, which is

expected considering that CGM sensors are designed to estimate interstitial glucose levels that approximate FBG concentrations. The histogram of carbohydrates is right-skewed, with the highest frequency of carbohydrate intake occurring in the lower range, specifically below 100 grams. The histogram of bolus insulin has a similar shape, with most insulin doses below 10 units. Most basal insulin doses fall within the range of 0.5 to 2 units per hour. However, there is a peak at 0 units per hour, likely reflecting instances when patients have temporarily disabled basal insulin delivery.

**Table 1.2** Descriptive statistics of BG, FBG, bolus, basal, and carbohydrates from the OhioT1DM dataset.

| | **BG [mg/dl]** | **FBG [mg/dl]** | **Bolus [u.]** | **Basal [u./h]** | **Carb. [g]** |
|---|---|---|---|---|---|
| count | 166533 | 4762 | 3733 | 165359 | 2168 |
| mean | 159.58 | 158.16 | 5.95 | 1.01 | 45.40 |
| std | 60.67 | 75.66 | 4.41 | 0.42 | 33.65 |
| min | 40.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| 25% | 113.00 | 107.00 | 2.60 | 0.73 | 21.75 |
| 50% | 152.00 | 152.00 | 4.80 | 0.98 | 39.000 |
| 75% | 197.00 | 203.00 | 8.50 | 1.25 | 60.00 |
| max | 400.00 | 586.00 | 25.00 | 2.34 | 450.00 |

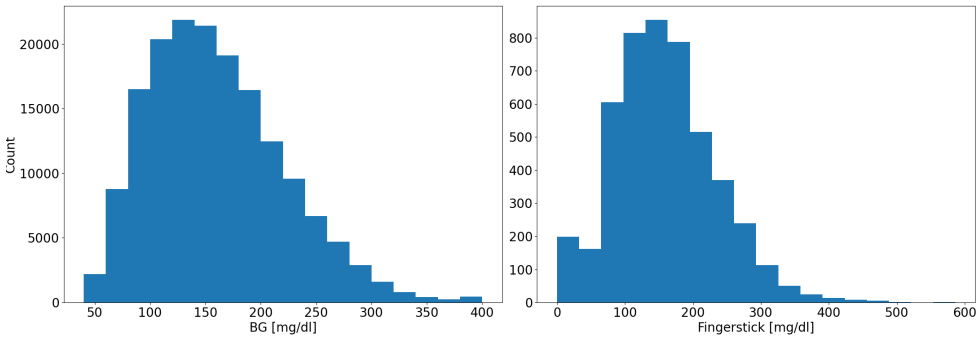



**Figure 1.2** Histograms of BG and FBG from the OhioT1DM dataset.

Figure 1.4 is a glance at a day's worth of data from a single patient with BG, carbohydrates, and bolus insulin displayed in the figure. Fluctuations in BG levels can be observed, particularly around meal times and at times when insulin is administered. Furthermore, it can be seen that there was a gap in BG measurements approximately between 12:00 and 18:00. Gaps in the BG measurements are quite common and are further explored in Table 1.3. Additionally, the BG spike observed after 18:00 does not appear to be preceded by any recorded carbohydrate consumption. This might suggest that the patient did not accurately log the meal.

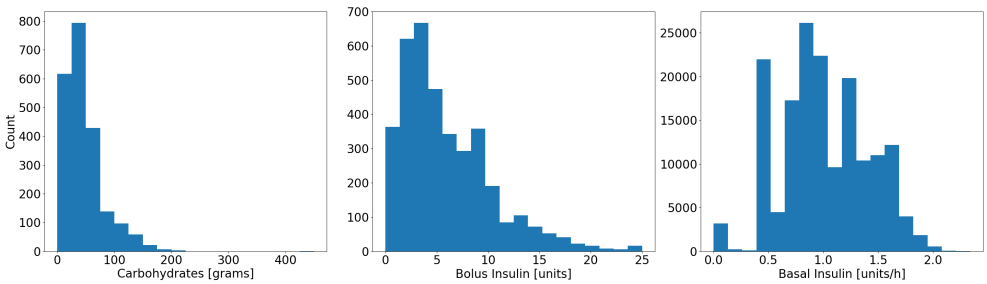



**Figure 1.3** Histograms of bolus, basal, and carbohydrates from the OhioT1DM dataset.

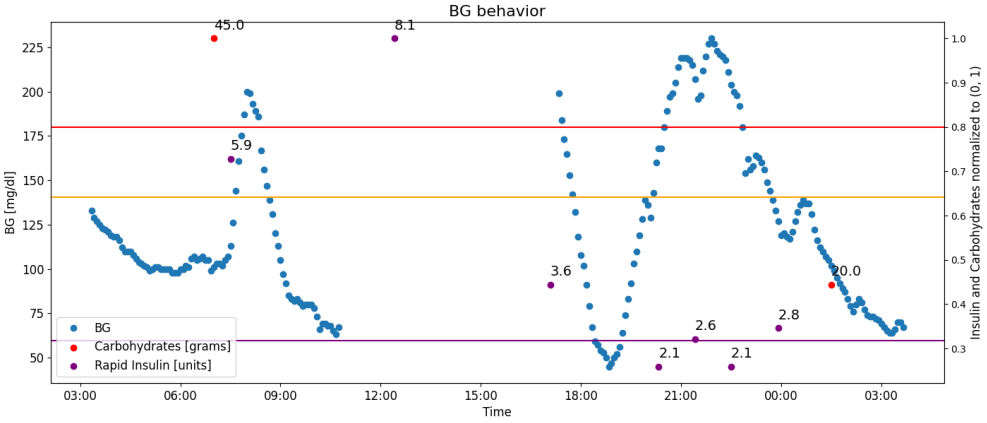



**Figure 1.4** Short-term BG behavior with purple line threshold for hypoglycemia, orange for euglycemia, and red for hyperglycemia.

Table 1.3 provides comprehensive data on mean BG gap durations, gap counts, and missing percentages across all patients and both cohorts.

The mean gap duration, gap count, and missing percentage of BG values show significant variation among patients. For example, in the training dataset, patient 588 exhibits the lowest missing percentage at 4.59%, whereas patient 596 has the highest at 23.88%. In the testing dataset, the missing percentages range from as low as 3.06% for patient 591 to as high as 40.72% for patient 552. These disparities suggest that using a straightforward approach of resampling and interpolating to handle missing values may not effectively represent the true dynamics of BG levels. A more detailed discussion on strategies for managing missing BG values can be found in Section 3.3.

Lastly, Figure 1.5 presents a series of violin plots illustrating the distribution of BG levels for individual patients. The body of each violin plot provides a mirrored density estimation showing the probability density of the data at different values, with the middle horizontal line in each plot indicating the mean. The shapes of the violins for both cohorts are generally similar. However, individual variations exist. A distribution displaying a long tail extending

**Table 1.3** Patient data on BG gaps, including mean gap duration, gap count, and missing percentages by patient and cohort.

| ID | Dataset | Cohort | Mean Duration [min] | Count | Missing % |
|---|---|---|---|---|---|
| 559 | test | 2018 | 169 | 11 | 12.59 |
| 563 | test | 2018 | 206 | 3 | 8.67 |
| 570 | test | 2018 | 80 | 9 | 4.69 |
| 575 | test | 2018 | 69 | 10 | 5.65 |
| 588 | test | 2018 | 227 | 2 | 3.09 |
| 591 | test | 2018 | 113 | 4 | 3.06 |
| 540 | test | 2020 | 111 | 8 | 5.54 |
| 544 | test | 2020 | 355 | 6 | 13.39 |
| 552 | test | 2020 | 798 | 10 | 40.72 |
| 567 | test | 2020 | 224 | 11 | 19.62 |
| 584 | test | 2020 | 115 | 15 | 11.02 |
| 596 | test | 2020 | 221 | 6 | 8.66 |
| 559 | train | 2018 | 157 | 42 | 10.75 |
| 563 | train | 2018 | 236 | 21 | 7.68 |
| 570 | train | 2018 | 162 | 20 | 7.00 |
| 575 | train | 2018 | 90 | 72 | 9.45 |
| 588 | train | 2018 | 238 | 10 | 4.59 |
| 591 | train | 2018 | 371 | 26 | 16.30 |
| 540 | train | 2020 | 198 | 30 | 9.83 |
| 544 | train | 2020 | 470 | 22 | 16.17 |
| 552 | train | 2020 | 234 | 44 | 19.16 |
| 567 | train | 2020 | 239 | 57 | 19.78 |
| 584 | train | 2020 | 98 | 59 | 8.29 |
| 596 | train | 2020 | 534 | 26 | 23.88 |

upward indicates a higher frequency of hyperglycemia episodes. Conversely, a long tail at the lower end of the distribution suggests more frequent occurrences of hypoglycemia.

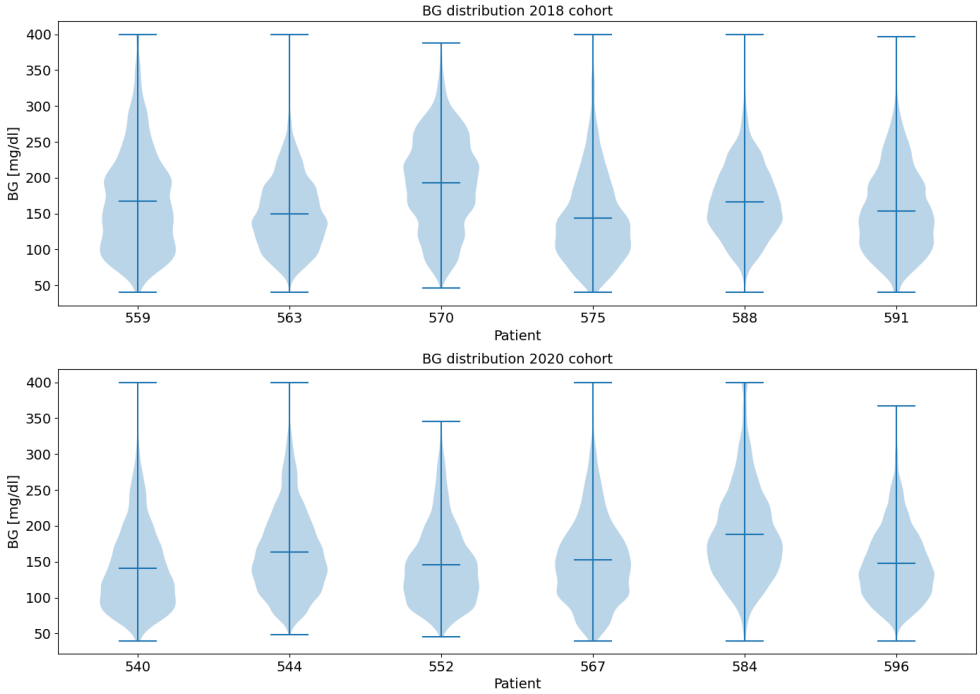



**Figure 1.5** Violin plot of BG for each patient.

# Chapter 2
# Related Work

## 2.1 Broad Overview

One of the earliest studies on BG prediction was conducted by Bremer and Gough in 1999 [18]. The researchers demonstrated that future BG levels could be predicted using only historical BG data. Ashenafi Zebene Woldaregay et al. mapped out the landscape of available literature on BG prediction in a comprehensive review [19]. The review gives an insight into the models used, prediction horizons, and achieved performance. Further, there is information about study subjects and inputs used for the models. The review considered peer-reviewed journals, articles, and conference proceedings published between 2000 and 2018.

Their findings show that the most commonly investigated prediction horizons range from 15 minutes to 2 hours. Figure 2.2 shows the most frequent combinations of features used to train models for BG prediction as reported in [19], with BG, Insulin, and Diet being the most frequent. Figure 2.1 showcases the most common types of models used for BG prediction according to [19]. It can be seen that most commonly, they are feed-forward neural networks, closely followed by recurrent neural networks (RNN) and then hybrid approaches. Most of the hybrid BG prediction models involve the hybridization of physiological (compartmental) models and different ML techniques.

Physiological models are typically used to convert raw features, such as insulin doses, carbohydrate intake, and potentially physical activity, into a format that should enhance the predictive performance of ML models. This process usually involves transforming these features from discrete to continuous. These continuous features aim to better represent the dynamics of insulin absorption into the circulation and the conversion of carbohydrates into readily available glucose for use by the body. For additional details, see Section 3.1.

Various other models were previously used for the BG prediction task,

including support-vector regression [20] and statistical approaches like ARIMA [21].

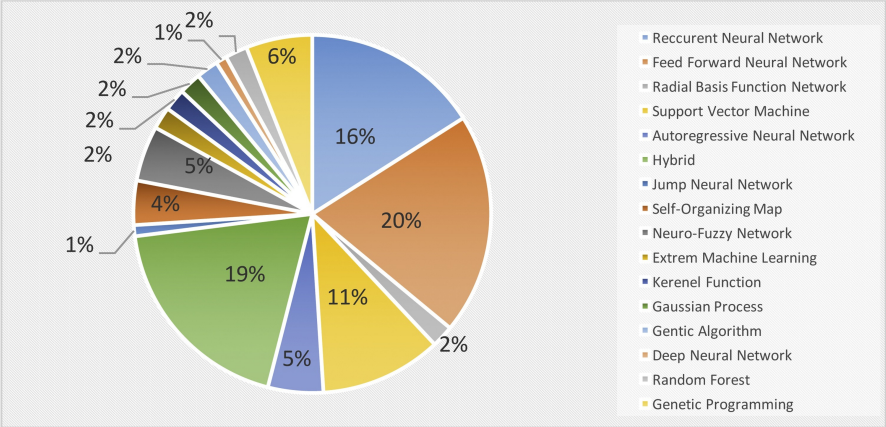



**Figure 2.1** Classes of ML techniques used for BG prediction. [19]

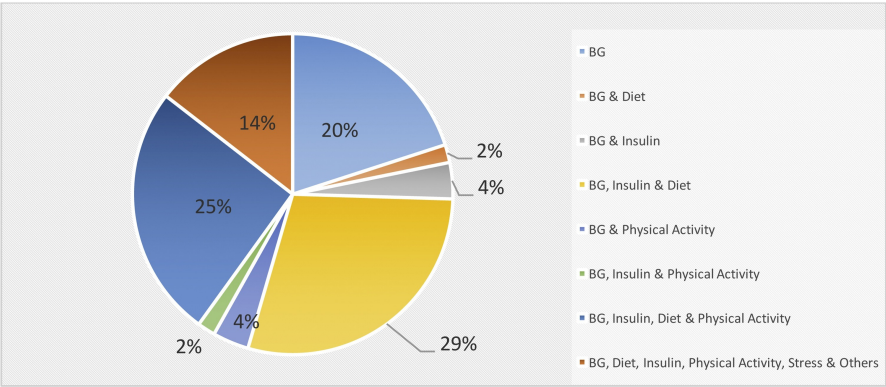



**Figure 2.2** Types of input features used for BG prediction. [19]

It is important to note that comparing prediction performance across different studies is challenging because model evaluations often do not utilize the same datasets. The predictability of BG dynamics can vary significantly among patients; for instance, individuals with routine daily activities and regular BG patterns may present less complex prediction tasks compared to those with more irregular lifestyles and BG fluctuations.

Furthermore, the methodologies and data transformations used in these studies can differ substantially. For example, for any time $t$ and $PH$, Georga et al. [22] excluded data windows from their dataset that contained any events such as food intake, insulin administration, or moderate to intense exercise within the time interval $[t, t + PH]$. This exclusion of training instances is intended to establish a more logical connection between input and output by not considering unpredictable future events. However, it also makes the data

less noisy, and it is harder to make predictions if such dropout is not used.

In another example, Rabby et al. [23] applied Kalman smoothing to the BG values across their entire dataset, effectively reducing noise and simplifying the prediction process. They reported results for both the smoothed and unsmoothed data, but it is crucial to carefully interpret these results, recognizing how such data pre-processing may influence the apparent prediction performance.

Given the difficulties in comparing prediction performance across different studies due to varying methodologies and datasets, this thesis will only compare with models developed using the OhioT1DM dataset. Additionally, careful attention has been given to accurately interpret the reported results to ensure that meaningful comparisons can be made with the selected studies.

## 2.2 Works utilizing the OhioT1DM dataset

This section introduces works that utilized the 2018 and/or 2020 editions of the OhioT1DM dataset.

### 2.2.1 Performance on OhioT1DM, 2018 edition

Chen et al. in [24] proposed a Dilated Recurrent Neural Network for a 30-minute BG prediction task, utilizing the 1-hour input containing CGMs, bolus insulin doses, and carbohydrate intake, achieving a mean root mean squared error (mean RMSE) of 19.04 mg/dl. Performance was evaluated on all test data points.

Martinsson et al. in [25] implemented an LSTM-based neural network that utilized the past 30 minutes of BG history to make BG prediction 30 and 60 minutes in the future, achieving a mean RMSE of 20.1 mg/dl and 33.2 mg/dl, respectively. Performance was evaluated only on test points where at least 12 previous consecutive measurements were available.

Rabby et al. in [23] implemented a stacked LSTM neural network, which utilized the past 2 hours of BG values, carbohydrate intake from the meal, insulin dose as a bolus, and 5-min aggregation of step count from the fitness band. The carbohydrate and insulin features were transformed into continuous variables instead of using the raw discrete samples. They further experimented with Kalman smoothing of the input and target BG time series. They report RMSE for both 30 and 60-minute PH. The RMSE achieved for raw BG input and output was 18.57 mg/dl and 30.32 mg/dl, respectively. The best RMSE for the Kalman smoothed BG was 5.89 mg/dl and 17.24 mg/dl, respectively. They mention that: "It is possible to evaluate the model at a certain point in time if there are at least 24 prior data points available (prior data points for 120 min)", which suggests that the performance was evaluated only on test points where at least 24 previous consecutive measurements were available, but it is not stated explicitly.

### 2.2.2 Performance on OhioT1DM, 2020 edition

The reported performance is always evaluated on all provided test data points of all patients from the 2020 edition of OhioT1DM. Some works also utilized the 2018 edition and/or other datasets for pre-training.

Zhu et al. in [8] proposed Generative Adversarial Networks (GAN) for BG prediction, utilizing 1.5 hours of historical data containing BG, carbohydrate intake, and bolus insulin. The first half of the cohort from the 2018 OhioT1DM edition was used for model pre-training. The achieved RMSE was 18.34 mg/dl and 32.21 mg/dl for 30-minute and 1-hour PH, respectively.

Rubin-Falcone et al. in [26] developed an N-BEATS model utilizing BG, finger stick glucose, bolus values, carbohydrate inputs, sine and cosine of time, and missingness indicators for BG values. They've built an ensemble of models, each using a different input length, and used the median as the ensemble prediction. The models were pre-trained on the 2018 version of the OhioT1DM dataset and the Tidepool dataset [9] and then fine-tuned per patient. The achieved mean RMSE was 18.22 mg/dl and 31.66 mg/dl for 30-minute and 1-hour PH, respectively. The authors further mention that without pre-training on OhioT1DM 2018 and Tidepool, the 30-minute PH performance was 18.87 mg/dl.

Daniels et al. in [27] proposed a multi-task learning approach by training a convolutional RNN (CRNN) with subject-specific layers. The model was trained on 2-hour-long inputs with BG, insulin bolus, carbohydrate intake, and reported exercise. The achieved RMSE was 19.79 mg/dl and 33.73 mg/dl for 30-minute and 1-hour PH, respectively.

# Chapter 3

# Pre-processing and Feature Engineering

The OhioT1DM dataset contains two files, train and test, for each patient. The training dataset is loaded and split into training and validation sets, with the first 80% of the data points used for training the models and the last 20% for validation.

The BG sensors used in the OhioT1DM study are expected to take glucose readings every 5 minutes. Occasionally, gaps in the readings may occur due to signal loss, malfunction, or the patient not wearing the sensor. Both train and test datasets are resampled to a 5-minute sampling frequency. The missing values for carbohydrates, insulin, and fingerstick are set to 0. Missing BG values are first marked missing and later interpolated/extrapolated when creating the input windows (vectors) for the models (see Section 3.3).

## 3.1 Physiological Models

### 3.1.1 Hovorka Model

Roman Hovorka et al. [28] developed a model predictive controller for use with subcutaneous insulin infusion with the aim of facilitating control during fasting conditions. The controller employed a nonlinear model of glucose kinetics. The model consists of a glucose subsystem (glucose absorption, distribution, and disposal), an insulin subsystem (insulin absorption, distribution, disposal) and an insulin action subsystem (insulin action on glucose transport, disposal, and endogenous production).

In this thesis, the glucose and insulin subsystem components are employed to transform discrete raw data of insulin and carbohydrate consumption into a continuous representation. This process is designed to reflect more accurately the dynamics of insulin absorption into the bloodstream and the conversion of

carbohydrates into readily available glucose in the body.

Hovorka et al. [28] describe the insulin absorption in the insulin subsystem as:

$$\begin{aligned} \frac{dS_1(t)}{dt} &= u(t) - \frac{S_1(t)}{t_{\max,I}} \\ \frac{dS_2(t)}{dt} &= \frac{S_1(t)}{t_{\max,I}} - \frac{S_2(t)}{t_{\max,I}} \end{aligned} \tag{3.1}$$

where $S_1$ and $S_1$ are a two-compartment chain representing absorption of subcutaneously administered short-acting (e.g., Lispro) insulin, $u(t)$ represents administration (bolus and infusion) of insulin, and $t_{\max,I}$ is the time-to-maximum insulin absorption.

Further, Hovorka et al. [28] describe the gut absorption rate $U_G(t)$ in the glucose subsystem, represented by a two-compartment chain with identical transfer rates $t_{\max,G}$ as:

$$U_G(t) = \frac{D_G \cdot A_G \cdot t \cdot e^{-t/t_{\max,G}}}{t_{\max,G}^2} \tag{3.2}$$

where $t_{\max,G}$ is the time-of-maximum appearance rate of glucose in the accessible glucose compartment, $D_G$ is the amount of carbohydrates digested, and $A_G$ is carbohydrate bioavailability.

### 3.1.2 Insulin Feature Engineering

The implementation of the Hovorka insulin model [28] transforms discrete bolus insulin samples into continuous features. The implementation is inspired by the work of Price [29]. A two-compartmental insulin model based on equation 3.1 estimates the amount of insulin in subcutaneous tissue (IST) and blood plasma as continuous features.

For each time step $t$, states $S_{1,t}$ and $S_{2,t}$ are maintained and represent the insulin in subcutaneous tissue and blood plasma respectively at time $t$. For each time $t$, the states are updated as follows:

$$\begin{aligned} S_{1,t} &= S_{1,t-1} + u(t) - \frac{S_{1,t-1}}{t_{\max,I}} \\ S_{2,t} &= S_{2,t-1} + \frac{S_{1,t}}{t_{\max,I}} - \frac{S_{2,t-1}}{t_{\max,I}} \end{aligned} \tag{3.3}$$

If no insulin $u(t)$ is administered at time $t$, the value of $u(t)$ will be 0. The parameter $t_{\max,I}$ of the time-to-max absorption of subcutaneously injected short-acting insulin was set to 100 minutes for all patients. Numerous values were explored for the time-to-max absorption, and it was seen that the models performed similarly well with values as low as 60 and as high as 180. Figure 3.1 shows an example of this insulin transformation. The IST feature was used to train the models.

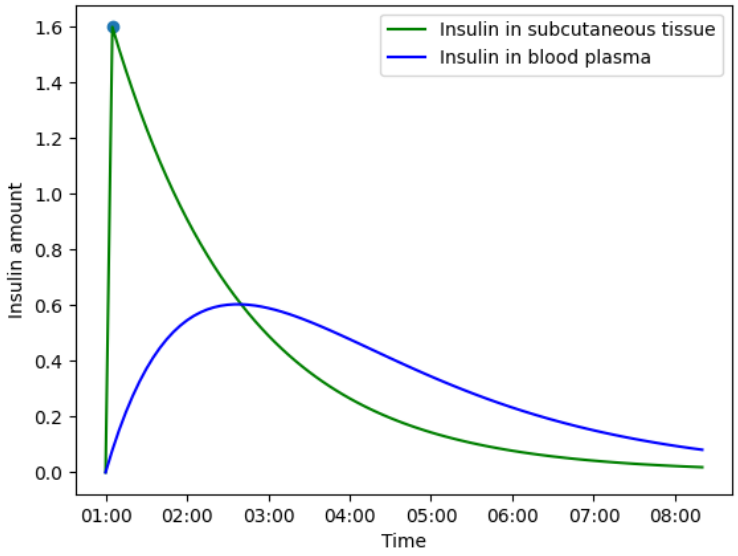



**Figure 3.1** Example of transforming discrete insulin samples into a continuous feature.

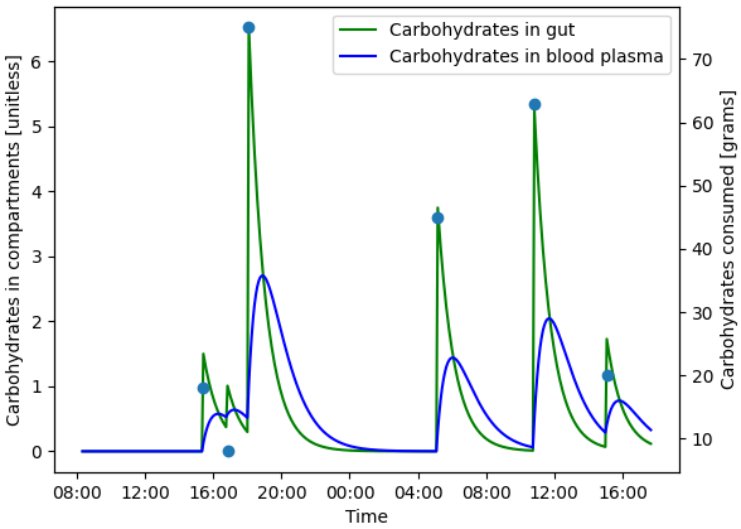



**Figure 3.2** Example of transforming discrete carbohydrate samples into a continuous feature.

### 3.1.3 Carbohydrates Feature Engineering

Similarly to insulin, discrete carbohydrate values are also transformed into a continuous feature using an implementation inspired by [29] of a two-compartmental meal model [28]. The model estimates the amount of glucose available in the gut and blood plasma. For each time step $t$, states $G_t$ and $P_t$ are maintained to represent the glucose in the gut and plasma, respectively. For each time $t$, the states are updated as follows:

$$\begin{aligned} G_t &= G_{t-1} + \frac{A_G \cdot D_{G,t}}{t_{\max,G}} - \frac{G_{t-1}}{t_{\max,G}} \\ P_t &= P_{t-1} + \frac{G_t}{t_{\max,G}} - \frac{P_{t-1}}{t_{\max,G}} \end{aligned} \tag{3.4}$$

where the $A_G$ as carbohydrate bioavailability was set to 1 and $D_{G,t}$ are the grams of carbohydrates consumed at time $t$, set to 0 if no carbohydrates were ingested. The $t_{\max,G}$ parameter for the time of maximum glucose rate of appearance was set to 60 minutes for all patients. This is the same value as the one used by Rabby et al. [23], although their carbohydrates model implementation is different. Figure 3.2 shows an example of this transformation. The carbohydrates in the gut (CG) feature was used to train the models.

## 3.2 Creating Windows

For any time $t$ and PH, the models should predict BG value at time $t + PH$ based on historical samples seen before time $t$. To facilitate this, a WindowGenerator class has been developed in Python to prepare windows of historical values as inputs and future BG levels as targets for the models. The key parameters of the WindowGenerator are as follows:

- Features: the feature to be used (BG, Carbohydrates, etc.).
- Input Width: Specifies the number of historical time steps used for each input window.
- PH: Indicates the future time point to predict, measured in time steps; for instance, a PH of 12 corresponds to a prediction one hour ahead, assuming a sampling frequency of five minutes.
- Batch Size: Determines the number of input-output window pairs per batch.

The windows are created with a stride equal to 1. So, for a total window size equal to 5, the window intervals $w_i$ will be $w_0 = [0, 4]$, $w_1 = [1, 5]$, ...

The shape of the input tensor is (BatchSize, InputWidth, NumberOfFeatures), where the BatchSize refers to the number of windows in a single batch, InputWidth is the number of historical samples used and NumberOfFeatures is how many dimensions are used to represent data in one time step (sample).

Moreover, the WindowGenerator class incorporates methods for handling data interpolation and the logic for dropping windows with insufficient data, as detailed in Section 3.3. This class is crucial for transforming the input time-series data into structured inputs and targets that models can effectively learn from.

## 3.3 Handling of Missing Blood Glucose Values

Missing BG entries are first substituted with FBG samples when available. In handling missing BG values within the training and validation datasets, sequences of missing BG data are linearly interpolated if the gap consists of up to six samples, equivalent to a duration of 30 minutes. Any missing samples that extend beyond this 30-minute window remain unfilled. This interpolation is applied across the entire dataset, affecting both the input features and the target outputs used during the model's training and validation phases.

Interpolation is not applied to the test dataset as a whole since this could introduce data leakage. Here, for each input and target, the following decision-making process is applied:

- if all input BG values are missing, the input-target pair is dropped,
- if the target BG value is missing, the input-target pair is dropped,
- else, that is, when the target value exists, and input BG has at least 1 value, the rest of the missing samples in the input are interpolated and forward and backward filled from the existing BG values in the input.

# Chapter 4
# Models

## 4.1 LSTM

A Long-Short-Term Memory (LSTM) recurrent neural network serves as the baseline model for comparisons with other models in this thesis. LSTM networks, developed by Sepp Hochreiter and Jürgen Schmidhuber [30], are distinguished by their ability to retain information over prolonged periods, improving upon the basic RNN design, which has issues with longer sequences. This characteristic makes them well-suited for machine translation, image captioning, and time-series forecasting tasks.

LSTM cell for each time $t$ takes in the input $x_t$ and maintains a cell state $C_t$ and hidden state $h_t$ (which is also the cell output). It uses a forget gate introduced in [31] and an input gate to update the cell state $C_{t-1}$ based on $x_t$ and $h_{t-1}$. Then, through the output gate, it updates the previous hidden state $h_{t-1}$ to create a new hidden state $h_t$ based on $C_t$, $x_t$ and $h_{t-1}$. For a schema of the LSTM cell, see Figure 4.1.

## 4.2 Transformer

The Transformer model, introduced by Vaswani et al. [33], is a recently popular neural network architecture based on an encoder-decoder structure. It advanced various tasks that require the processing of sequential data. The Transformer's core mechanism, known as attention, allows it to handle data sequences in parallel, contrary to the sequential processing of traditional RNNs. This capability significantly improves training efficiency and has led to impressive performance gains in areas such as natural language processing, where it powers innovations in machine translation, text summarization, and language generation.

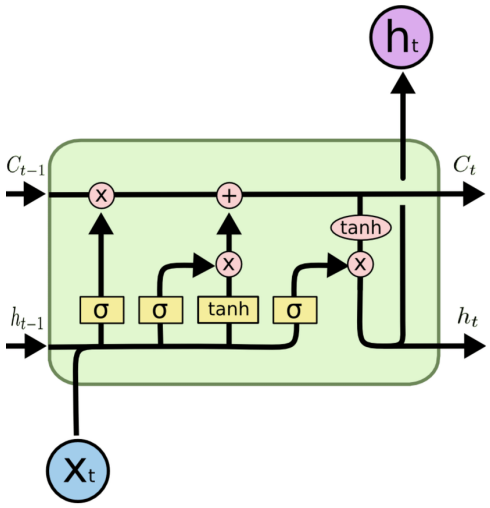



**Figure 4.1** LSTM cell [32].

### 4.2.1 Attention Mechanism

The attention mechanism is a main component of the Transformer architecture. It operates on the principle of dynamically focusing on different parts of the input sequence, thereby allowing the model to learn contextual relationships between features, regardless of their position in the sequence. Unlike previous architectures that processed input data in a fixed order, the attention mechanism provides a flexible way to weigh the importance of each part of the input data, facilitating more effective learning of dependencies.

Vaswani et al. [33] describe attention as a function that maps a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. In practice, the attention function is computed on a set of queries simultaneously, packed together into a matrix $\mathbf{Q}$. The keys and values are also packed together into matrices $\mathbf{K}$ and $\mathbf{V}$. The matrix of outputs is computed as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\mathbf{T}}}{\sqrt{d_k}}\right)\mathbf{V} \tag{4.1}$$

### 4.2.2 Architecture

The Transformer's architecture comprises two main modules: the encoder and the decoder. The encoder consists of a stack of layers that process the input sequence, each employing self-attention and position-wise feed-forward networks. The decoder, similarly layered, includes both self-attention and

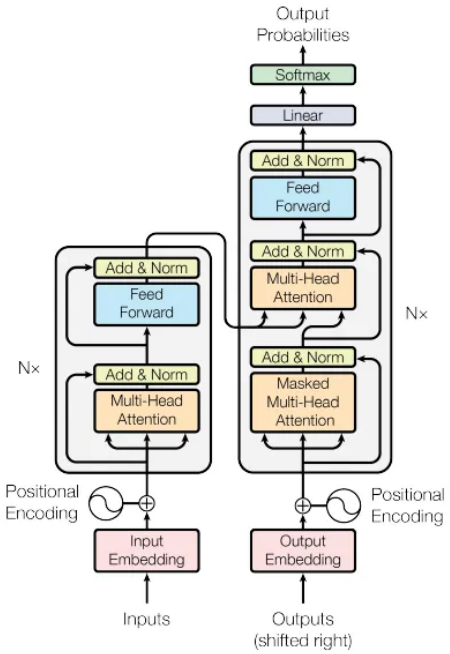



**Figure 4.2** The Transformer - model architecture (adopted from [33]).

encoder-decoder attention mechanisms. Self-attention in the decoder ensures the generation of coherent output sequences, while the encoder-decoder attention layers allow the decoder to focus on relevant parts of the input sequence. This two-part structure is adept at handling a wide range of sequence-to-sequence tasks. The Transformer architecture schema can be seen in Figure 4.2.

## 4.3 Time Series Transformer

The Time Series Transformer is a vanilla encoder-decoder Transformer for time series forecasting. The implementation used in this thesis is from Hugging Face [34]. It is a basic transformer with a distribution head on top of it, which can be used for time-series forecasting. It is a probabilistic forecasting model, not a point forecasting model. This means that the model learns a distribution from which one can sample.

The Time Series Transformer consists of 2 blocks: an encoder, which takes time series values as input, and a decoder, which predicts time series values into the future. During training, one needs to provide pairs of both past and future time-series values to the model. [34]

The whole future time series in the time interval $[t, t + PH]$ is provided to the models during training, and the model outputs a sequence of PH values. Thus, the loss is also computed across the entire output sequence. The results

reported later in Chapter 5 are only evaluated on the last predicted value in the sequence at $t+PH$. Since the model learns a distribution, the loss function for training is negative log-likelihood.

## 4.4 Informer

Informer is a transformer-based architecture developed by Zhou et al. [35]. The implementation of Informer used in this thesis is also from Hugging Face [36]. It introduces a probabilistic attention mechanism and provides a sparse transformer, thus mitigating the quadratic computing and memory requirements of vanilla attention. [36]

Zhou et al. [35] explain that the ProbSparse Self-attention proposed by them allows each key to only attend to the $u$ dominant queries according to a query sparsity measurement. Thus, the $\mathbf{Q}$ matrix from 4.1 will contain the top-u queries under the sparsity measurement, controlled by a constant sampling factor $c$.

Except for the addition of the sampling factor $c$, the parameters of the Informer remain the same as those of the Time Series Transformer.

## 4.5 Legendre Memory Unit

Legendre Memory Unit (LMU), introduced in [37], is a relatively less known type of RNN architecture designed to process time-series data efficiently. It leverages the mathematical properties of Legendre polynomials to create a fixed-size memory cell, enabling it to achieve high precision in capturing and representing temporal information over sequences with long-range dependencies.

In general, the LMU cell consists of two parts: a memory component (decomposing the input signal using Legendre polynomials as a basis) and a hidden component (learning nonlinear mappings from the memory component) [38], for more, see 4.5.1 and 4.5.3.

### 4.5.1 Memory Cell

Voelker et al. [37] explain that the memory cell orthogonalizes the continuous-time history of its input signal, $u(t) \in \mathbb{R}$, across a sliding window of length $\theta \in \mathbb{R}_{>0}$. The cell is derived from the linear transfer function for a continuous-time delay, $F(s) = e^{-\theta s}$, which is best approximated by $d$ coupled ordinary differential equations (ODEs):

$$\theta \dot{\mathbf{m}}(t) = \mathbf{A}\mathbf{m}(t) + \mathbf{B}u(t) \tag{4.2}$$

where $\mathbf{m}(t) \in \mathbb{R}^d$ is a state-vector with $d$ dimensions. The ideal state-space matrices, $(\mathbf{A}, \mathbf{B})$, are derived through the use of Padé [39] approximants [40]:

$$\begin{aligned} \mathbf{A} = [a_{ij}] \in \mathbb{R}^{d\times d}, \quad a_{ij} &= \begin{cases} (2i+1)(-1)^i & \text{for } i < j, \\ (2j+1)(-1)^{i-j+1} & \text{for } i \geq j. \end{cases} \\ \mathbf{B} = [b_i] \in \mathbb{R}^{d\times 1}, \quad b_i &= (2i+1)(-1)^i, \quad i, j \in [0, d-1]. \end{aligned} \tag{4.3}$$

The key property of this dynamical system is that $\mathbf{m}$ represents sliding windows of $u$ via the Legendre [41] polynomials up to degree $d-1$:

$$\begin{aligned} u(t-\theta') &\approx \sum_{i=0}^{d-1} P_i\left(\frac{\theta'}{\theta}\right) m_i(t), \quad 0 \leq \theta' \leq \theta, \\ P_i(r) &= (-1)^i \sum_{j=0}^{i} \binom{i}{j}\binom{i+j}{j}(-r)^j \end{aligned} \tag{4.4}$$

where $P_i(r)$ is the $i^{th}$ shifted Legendre polynomial [42]. This gives a unique and optimal decomposition, wherein functions of $\mathbf{m}$ correspond to computations across windows of length $\theta$, projected onto $d$ orthogonal basis functions. For a visual representation of Legendre polynomials, see 4.3.

#### 4.5.1.1 Discretization

The matrices $\mathbf{A}$ and $\mathbf{B}$ describing the ideal state-space model need to be discretized, which in this case means mapping the equations onto the memory of an RNN. Voelker et al. [37] explain that for a given RNN memory $\mathbf{m}_t \in \mathbb{R}^d$, and given some input $u_t \in \mathbb{R}$, indexed at discrete moments in time, $t \in \mathbb{N}$:

$$\mathbf{m}_t = \tilde{\mathbf{A}}\mathbf{m}_{t-1} + \tilde{\mathbf{B}}u_t \tag{4.5}$$

where $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}})$ are the discretized matrices provided by the ODE solver for some time-step $\Delta t$ relative to the window length $\theta$. For instance, Euler's method supposes $\Delta t$ is sufficiently small:

$$\tilde{\mathbf{A}} = (\Delta t/\theta)\mathbf{A} + \mathbf{I}, \quad \tilde{\mathbf{B}} = (\Delta t/\theta)\mathbf{B}. \tag{4.6}$$

### 4.5.2 Layer Design

In [37], it is described that the LMU takes an input vector, $\mathbf{x}_t$, and generates a hidden state, $\mathbf{h}_t \in \mathbb{R}^n$. Each layer maintains its own hidden state and memory vector. The state mutually interacts with the memory, $\mathbf{m}_t \in \mathbb{R}^d$, in order to compute nonlinear functions across time while dynamically writing to memory. Similar to the NRU [43], the state is a function of the input, previous state, and current memory:

$$\mathbf{h}_t = f(\mathbf{W}_\mathbf{x}\mathbf{x}_t + \mathbf{W}_\mathbf{h}\mathbf{h}_{t-1} + \mathbf{W}_\mathbf{m}\mathbf{m}_t) \tag{4.7}$$

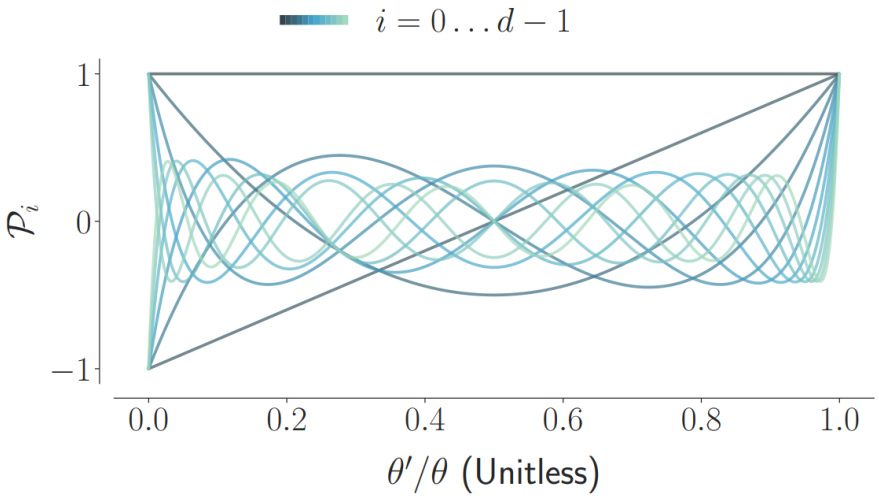



**Figure 4.3** Shifted Legendre polynomials ($d = 12$).

The memory of the LMU represents the entire sliding window of input history as a linear combination of these scale-invariant polynomials. Increasing the number of dimensions supports the storage of higher-frequency inputs relative to the time-scale. [37]

where $f$ is some chosen nonlinearity (e.g., tanh) and $\mathbf{W_x}, \mathbf{W_h}, \mathbf{W_m}$ are learned kernels. Note this decouples the size of the layer's hidden state ($n$) from the size of the layer's memory ($d$) and requires holding $n + d$ variables in memory between time-steps. The input signal that writes to the memory (via equation 4.5) is:

$$u_t = \mathbf{e_x}^\top \mathbf{x}_t + \mathbf{e_h}^\top \mathbf{h}_{t-1} + \mathbf{e_m}^\top \mathbf{m}_{t-1} \tag{4.8}$$

where $\mathbf{e_x}$, $\mathbf{e_h}$, $\mathbf{e_m}$ are learned encoding vectors. Intuitively, the kernels $\mathbf{W}$ learn to compute nonlinear functions across the memory, while the encoders $\mathbf{e}$ learn to project the relevant information into the memory. For a schematic of the LMU layer design, see Figure 4.4.

### 4.5.3 Hidden Component

The LMU architecture includes a hidden component that is responsible for learning nonlinear relationships from the memory component and producing the state vector $\mathbf{h}_t \in \mathbb{R}^n$. According to the Nengo LMU implementation documentation [38], this hidden component can take various forms, such as a standard RNN cell, LSTM cell, and Gated Recurrent Unit (GRU) cell. Additionally, it can also be a feed-forward layer or None (empty) to create a memory-only LMU. The number of units used in the hidden cell determines the dimensionality of $\mathbf{h}_t$.

The Standard/Simple RNN cell can be seen in Figure 4.5. GRU, as seen in

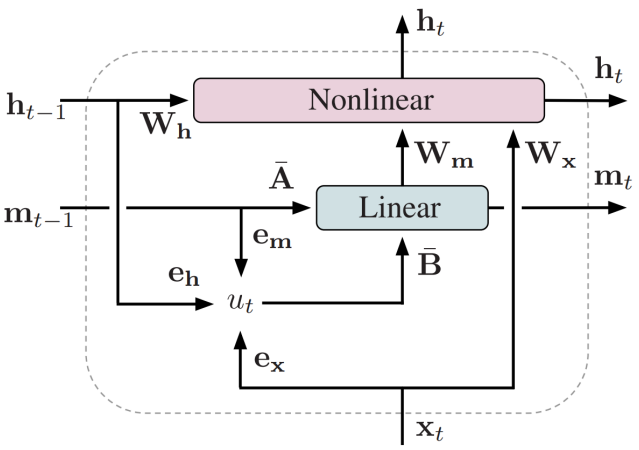



**Figure 4.4** Time-unrolled LMU layer.

An $n$-dimensional state vector ($\mathbf{h}_t$) is dynamically coupled with a $d$-dimensional memory vector ($\mathbf{m}_t$). The memory represents a sliding window of $u_t$, projected onto the first $d$ Legendre polynomials. [37]

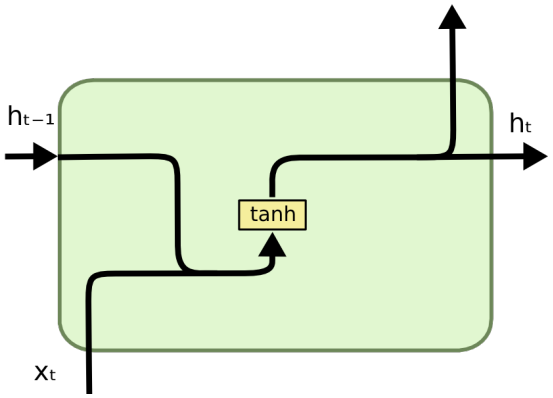



**Figure 4.5** Standard/Simple RNN cell [32].

Figure 4.6 is similar to the LSTM but it combines the forget and input gates into a single "update gate.", with some additional changes.

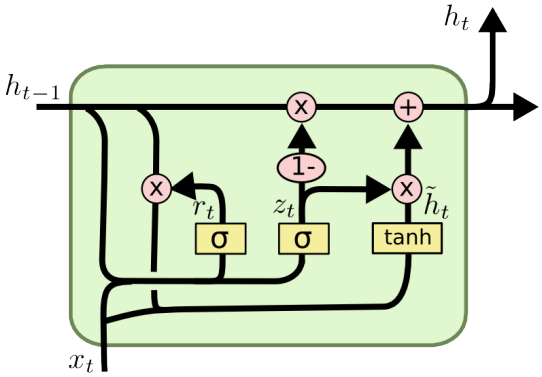



**Figure 4.6** GRU cell [32].

# Chapter 5

# Model Development and Experiments

This thesis examines three distinct model architectures: the Time Series Transformer, the Informer, and the LMU. This chapter outlines the procedures and experiments conducted to apply, evaluate, and fine-tune these models for the task of BG prediction.

## 5.1 Model Evaluation Strategy

Two metrics are used to evaluate the prediction performance of models: Root Mean Squared Error (RMSE) and, for assessing clinical accuracy, Clarke Error Grid Analysis (CEGA) [44].

For a series of BG measurements $Y$ and BG predictions $\hat{Y}$, both having an equal length $n$, RMSE is defined as

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2} \tag{5.1}$$

Each model is evaluated by calculating the RMSE for each patient and computing the mean RMSE as the mean of the RMSEs of all patients. This results in each patient's RMSE having the same weight in the final mean RMSE, even though the number of test samples is different between patients.

CEGA is widely used to assess the clinical accuracy of BG prediction models. It presents a scatterplot that compares reference BG values against predicted BG values and categorizes the results into five zones:

- Zone A includes predicted BG values with deviations no greater than 20% from actual BG measurements, representing ideal accuracy.
- Zone B encompasses predictions that deviate beyond 20% but would not lead to incorrect treatment.

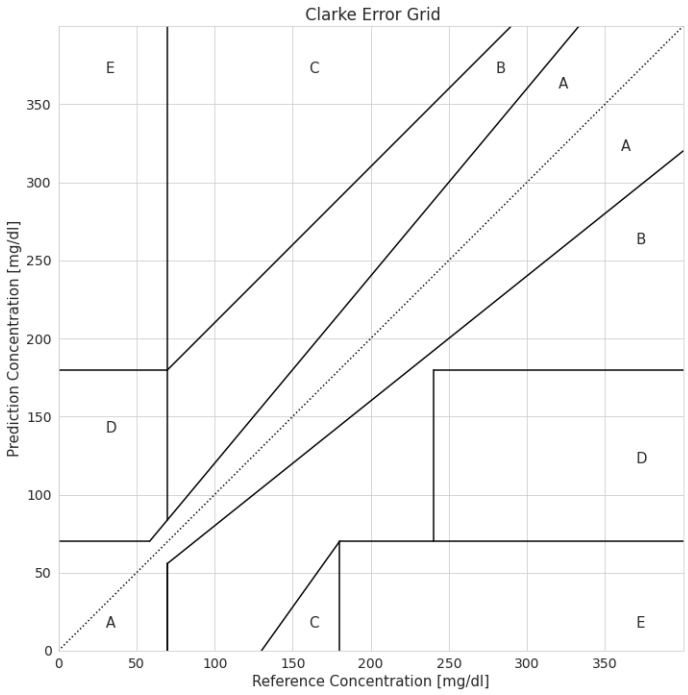



**Figure 5.1** Clarke error grid.

- Zone C includes points that might prompt unnecessary treatment.
- Zone D comprises points indicating a potentially dangerous failure to detect hypoglycemia or hyperglycemia.
- Zone E contains clinical errors where hyperglycemia is confused for hypoglycemia and vice versa.

The goal is for all predictions to fall within Zone A, with Zone B representing acceptable discrepancies and the remaining zones containing as few points as possible. A visual example of CEGA is provided in Figure 5.1. CEGA is performed on the best model in Chapter 6.

## 5.2 Model Training

This section explains the model training setup used for the final evaluation of models, as presented in Chapter 6 and for the hyper-parameter tuning.

For the 2018 OhioT1DM edition, models were trained using data from all patients in the 2018 cohort. Models for the 2020 edition were trained using both the train and test data from the 2018 edition and the training datasets from all patients in the 2020 cohort. All models were trained using the same input features: BG, FBG, insulin in subcutaneous tissue, and carbohydrates

in the gut. The rationale behind selecting these features is detailed in Section 5.4.1. The PH evaluated were 30 minutes and 1 hour.

All models were designed to predict the change $G_{t+\text{PH}} - G_t$ of the BG values at $t$ and $t+\text{PH}$. The rationale for this choice is written out in Section 5.4.3. The final BG prediction of a model represented by a function $f_\Delta(\vec{x}_{t_1}, \ldots, \vec{x}_{t_n})$ is then given by $\hat{G}_{t+\text{PH}} = f_\Delta(\vec{x}_{t_1}, \ldots, \vec{x}_{t_n}) + G_t$. In the case of Time Series Transformer and Informer, the models are trained to predict the change $G_{t+i} - G_t$ for each sample at time $i$ in the interval $[t, t + \text{PH}]$.

### 5.2.1 LSTM and LMU Training

All of the models were trained using Adam optimizer [45] with a learning rate set to $10^{-3}$, maximum of 300 epochs, and early stopping with the patience of 35, meaning the model training was stopped if validation loss did not improve in 35 epochs. The batch size was 256. The loss function used was mean squared error (MSE), defined as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{5.2}$$

for a series of BG measurements $Y$ and BG predictions $\hat{Y}$, both having an equal length $n$.

Furthermore, a learning rate reducer was used, which reduces the learning rate during the training by a factor of $10^{-1}$ when validation loss stagnates for 10 epochs. Model checkpoints were implemented to preserve the model with the best validation loss in each training.

### 5.2.2 Transformers Training

All models were trained using the Adam optimizer with an initial learning rate of $10^{-3}$, a maximum of 550 epochs, and early stopping implemented with a patience of 30 epochs. The learning rate was reduced to $10^{-4}$ after 50 epochs or if the validation loss did not improve for 30 epochs, and further reduced to $10^{-5}$ after 300 epochs using the same criteria. The batch size for training was set to 256.

An important distinction of the Transformer architecture from LSTM and LMU models is its requirement for positional encodings of individual samples to provide the model with the sequence order of the samples. In this context, the implementations for the Time Series Transformer [34] and Informer [36] include an interface to incorporate time features as positional encodings. Unlike the Transformer architecture, where encodings are learned, the time features act as positional encodings in these models. The time features used were Hour, Minute, and Age. The 'Age' feature assigns an index to each sample from 0 for the first to $n-1$ for the last n-th sample.

## 5.3 Hyper-parameter Tuning

The models' hyper-parameters were tuned using the Hyperband algorithm [46], introduced in 2018. Hyperband is a relatively recent method for hyper-parameter tuning that leverages adaptive resource allocation and early stopping to identify high-performing models quickly. This algorithm is initiated by training a large pool of models for a limited number of epochs and then progresses only the top-performing half to subsequent rounds.

### 5.3.1 LMU Tuning

LMUs have several hyper-parameters that can be set: hidden-to-memory connection (HM), memory-to-memory learnable connection (MM), input-to-hidden connection (IH), memory dimension, number and type of hidden units, dropout and recurrent dropout rate, and the order, which is the number of degrees in the transfer function of the linear time-invariant system used to represent the sliding window of history.

The type of hidden units in the LMU cells was chosen to be LSTM, which was shown to yield the best performance (see 5.4.2). The memory dimension was set at 4 to match the dimensionality of the input, while the remaining hyper-parameters were subject to tuning.

This tuning process was conducted separately for different dataset editions and PHs. Specifically, for the 30-minute PH, tuning was started with various input lengths: 30, 45, 60, and 120 minutes. The best validation loss was achieved with the 30-minute long input, which was subsequently used for tuning models for 60-minute PH. Table 5.1 shows the final best values for these hyper-parameters.

Additional manual experiments involved stacking two LMU layers rather than using a single layer; however, this configuration did not improve performance.

**Table 5.1** Best hyper-parameters for LMU models tuned on 2018 and 2020 OhioT1DM cohorts and 30-minute and 60-minute PH.

| Cohort | PH | order | HM | MM | IH | units | dropout | rec. drop. |
|---|---|---|---|---|---|---|---|---|
| 2018 | 30 | 52 | True | False | False | 72 | 0.1 | 0 |
| 2018 | 60 | 54 | False | True | False | 144 | 0.1 | 0 |
| 2020 | 30 | 60 | True | False | False | 72 | 0.1 | 0.2 |
| 2020 | 60 | 64 | False | False | True | 156 | 0.1 | 0.1 |

### 5.3.2 LSTM Tuning

The hyper-parameters tuned for the LSTM were the number of units and the dropout rate. Two distinct LSTM architectures were tuned: one with a single

LSTM layer and another with two stacked LSTM layers. The dropout was added after the LSTM layer, and for the stacked variant, it was also added between the two LSTM layers. This tuning process was run separately for different dataset editions and PHs. The input length was 30 minutes.

The tuning results indicated that the single-layer LSTM architecture performed worse than the stacked architecture. The optimal hyper-parameter values for the stacked architecture, which yielded the best performance, are shown in Table 5.2. The number of units was the same for both LSTM layers in the stacked architecture. Dropout 1 refers to the dropout applied between LSTM layers, while dropout 2 refers to the one applied between the last LSTM layer and the output layer.

**Table 5.2** Best hyper-parameters for LSTM models tuned on 2018 and 2020 OhioT1DM cohorts and 30-minute and 60-minute PH.

| Cohort | PH | units | dropout 1 | dropout 2 |
|---|---|---|---|---|
| 2018 | 30 | 240 | 0 | 0.2 |
| 2018 | 60 | 250 | 0.1 | 0.1 |
| 2020 | 30 | 200 | 0 | 0 |
| 2020 | 60 | 140 | 0 | 0.5 |

### 5.3.3 Transformers Tuning

The tuned hyper-parameters of the transformer models are [36], [34]:

- <ffn_dim> Dimension of the feed-forward layer of the decoder/encoder.
- <attention_heads> Number of attention heads for each attention layer in the encoder/decoder.
- <layers> Number of encoder/decoder layers.
- <d_model> Dimensionality of the transformer layers.
- <dropout> The dropout probability for all fully connected layers in the encoder and decoder.
- <layerdrop> The dropout probability for the attention and fully connected layers for each encoder/decoder layer.
- <attention_dropout> The dropout probability for the attention probabilities.
- <activation_dropout> The dropout probability used between the two layers of the feed-forward networks.

- <sampling_factor> Applicable to Informer only, controls the reduced query matrix **Q** input length.

Where *ffn_dim*, *attention_heads*, *layers*, and *layerdrop* parameter values were passed to both the encoder and decoder of the transformer architecture.

Hyper-parameter tuning for the Informer model was initially conducted for the 2020 cohort with a PH of 30 minutes. Tuning was started for two input lengths: 120 minutes and 30 minutes, with a 30-minute input length achieving better results. The best parameters are listed in Table 5.3. Due to the model's poor performance, as indicated in Table 6.1, further tuning for a 60-minute PH and the 2018 cohort was halted to focus efforts on more promising architectures, particularly the LMU.

For completeness, the Informer model was also trained for a 60-minute PH for the 2020 cohort and both 30-minute and 60-minute PHs for the 2018 cohort. The Time Series Transformer was also trained for both cohorts and PHs. All of these models shared the same hyper-parameter setting as per Table 5.3, except Time Series Transformer, for which the parameter *sampling_factor* was omitted since it does not apply to this model. The validation RMSE for these models is presented in Table 6.1.

**Table 5.3** Best hyper-parameters for Informer tuned on 2020 OhioT1DM cohort and 30-minute PH.

| parameter | value |
|---|---|
| ffn_dim | 32 |
| attention_heads | 2 |
| layers | 2 |
| d_model | 128 |
| dropout | 0.1 |
| layerdrop | 0.1 |
| attention_dropout | 0.1 |
| activation_dropout | 0.1 |
| sampling_factor | 5 |

## 5.4 Ablation Study

This section outlines the steps taken to enhance the models developed in this thesis, including dataset pre-processing, feature engineering, and the selection of model parameters. Given that LMUs demonstrated superior performance compared to Transformer architectures, as shown in Table 6.1, the analysis focuses on LMUs.

The method involves first introducing the highest-performing LMU model and then systematically altering specific parameters and training setup to evaluate the impact of these changes. The ablation study focuses on a PH of 30

minutes using the 2018 dataset for simplicity and due to constraints on computational resources.

The best-performing model for predicting BG over a 30-minute PH was an LMU trained on 30-minute-long input data incorporating BG, FBG, IST, and CG features. The LMU model employed LSTM as its hidden unit, and its parameters were configured according to those listed in Table 5.1. The model was designed to predict the BG change: $G_{t+\mathrm{PH}} - G_t$. This model achieved a validation RMSE of **20.93 ± 0.03 mg/dl** and will be referred to as *BestLMU* in subsequent discussions.

See Table 5.4 for the validation RMSE associated with individual steps in the ablation study.

### 5.4.1 Effects of Features

This section examines the impact of individual features and feature engineering on the model's performance.

#### Compartmental Model-derived Features

First, the effect of using transformed insulin and carbohydrates features: "insulin in the subcutaneous tissue" (IST) and "carbohydrates in the gut" (CG) is assessed. These features are created based on the output from the compartmental models, discussed in Section 3.1. When *BestLMU* was trained using the raw Carbohydrates and Bolus Insulin features instead of the feature from compartmental models, the validation RMSE increased to 21.18 mg/dl, indicating that the usage of compartmental model-derived features leads to more accurate predictions.

The best input length for the models utilizing the compartmental features was observed to be 30 minutes. In the conducted experiments, longer input windows did not lead to better performance. Surprisingly, sometimes, they produced worse performance of the models. It is suspected that the feature engineering applied to insulin and carbohydrates might have helped decrease the input length needed. This assumption is made because the transformed carbohydrate and insulin features effectively stretch out the influence of insulin and carbohydrates in time. For example, a model with a 30-minute input window may still see the effects of insulin taken more than 8 hours ago.

#### Fingerstick Blood Glucose

Next, the removal of the FBG feature was explored. Training *BestLMU* without the FBG feature resulted in a validation RMSE of 21.04 mg/dl, suggesting that including FBG slightly enhances model performance.

#### Basal Insulin

The incorporation of basal insulin was also tested. The basal insulin, described as the number of insulin units released by the pump per hour, was first converted into insulin units per 5 minutes to match the sampling frequency of the time series. The basal insulin was then added to the bolus insulin, creating a new insulin feature that captured the effects of both. The compartmental model was then applied to this feature to create the IST feature. The RMSE achieved was 21.06 mg/dl, suggesting that adding basal insulin does not help the model.

#### Physical Activity

Lastly, the potential benefits of incorporating physical activity data were considered. Inconsistent data types between cohorts presented challenges: the 2018 cohort data included step count, while the 2020 cohort data featured acceleration measures. Step count was explored as an addition to the training features, and it led to a comparable performance on the 2018 dataset, with an RMSE of 20.89 mg/dl. Due to the inconsistencies and lack of substantial performance enhancement, physical activity data were excluded from the models developed for both cohorts.

### 5.4.2 Choice of Hidden Unit in LMU

LSTM, GRU, and standard RNN cells were assessed as potential hidden units for an LMU network. The *BestLMU* model was trained five times with each type of cell. The standard RNN variant achieved a validation RMSE of 21.01 ± 0.06 mg/dl, and the GRU variant recorded an RMSE of 20.98 ± 0.03 mg/dl. These results suggest that LSTM cells provide the best performance, while GRU and standard RNN cells, though comparable, consistently resulted in slightly higher errors across multiple training runs.

### 5.4.3 Choice of Model Output

Drawing inspiration from studies like Zhu et al. [8], the approach of predicting the change in BG levels, $G_{t+\mathrm{PH}} - G_t$ was explored, instead of directly forecasting the BG at future time t + PH.

The *BestLMU* model was trained three times to predict BG levels at $t + PH$ directly. This approach resulted in a validation RMSE of 25.79 ± 0.22 mg/dl. Interestingly, removing the 0.1 dropout rate in the LMU network improved validation RMSE, reducing it to 21.73 ± 0.07 mg/dl. Despite this improvement, the performance remained inferior to the RMSE achieved by models predicting BG change.

### 5.4.4 Effect of Training on Combined Datasets

Including additional patients in the training dataset has been shown to improve the model's overall performance. This improvement was illustrated, for example, by Rubin-Falcone et al. [26]. Specifically, they pre-trained their model using data from both the 2018 OhioT1DM cohort and the Tidepool dataset, then fine-tuned it using data from the 2020 OhioT1DM cohort.

To test this finding, two models for PH of 30 minutes were developed: one trained exclusively on data from the 2020 cohort and another trained on data from both the 2018 and 2020 cohorts. Inputs to the models were BG, FBG, IST, and CG. The models used the optimal hyper-parameters from Table 5.1 for the PH of 30 minutes and cohort 2020.

The validation RMSE was calculated using data only from the 2020 cohort. The model trained solely on the 2020 cohort achieved a validation RMSE of 21.76 mg/dl, whereas the model trained on both cohorts achieved a slightly better validation RMSE of 21.66 mg/dl.

Furthermore, training on the combined datasets also showed improved performance on the 2018 cohort. However, studies reporting performance on the 2018 cohort did not incorporate the 2020 data, as it was not available at the time. For fair comparisons, models assessed on the 2018 cohort data were exclusively trained on that specific dataset.

### 5.4.5 Optimizers

Lion optimizer [47], published recently in 2023, is a stochastic-gradient-descent method that uses the sign operator to control the magnitude of the update, unlike other adaptive optimizers such as Adam that rely on second-order moments. It has been shown to improve the accuracy of various ML models and is examined here as an alternative to the Adam optimizer. This section describes experiments conducted using the Lion optimizer to train the LMU model and the LSTM baseline.

Based on their experience, the authors report that a suitable learning rate for Lion is typically 3-10 times smaller than that for Adam [47]. For this reason, three initial learning rate settings were picked $\frac{1}{3} \cdot 10^{-3}$, $\frac{1}{5} \cdot 10^{-3}$ and $\frac{1}{10} \cdot 10^{-3}$.

The *BestLMU* model trained with Lion and learning rate $10^{-3}$ divided by factors 3, 5, and 10 achieved a validation RMSE of 21.01, 20.98, and 20.77 mg/dl, respectively. These results initially suggested that Lion could slightly outperform Adam when using a $10^{-4}$ learning rate. However, three additional training runs under the same settings did not reproduce these RMSEs, resulting in RMSEs of 20.92, 20.94, and 20.94 mg/dl. Lion thus failed to yield models with lower RMSE consistently.

The effect of using the Lion optimizer for training was further explored for the baseline LSTM architecture. The Lion optimizer was observed to yield

models with performance comparable to those trained using Adam.

**Table 5.4** Ablation Results for different training setups, RMSE is evaluated on the validation dataset.

| Training setup | RMSE [mg/dl] |
|---|---|
| Raw Insulin and Carbohydrate features | 21.18 |
| Removal of FBG feature | 21.04 |
| Addition of basal insulin feature | 21.06 |
| Addition of step count feature | 20.89 |
| Simple RNN as hidden LMU cell | 21.01 |
| GRU as hidden LMU cell | 20.98 |
| Predicting BG instead of BG change | 21.73 |
| *BestLMU* model | 20.93 |

## 5.5 LMU with Simple RNN Cell

LMU with a simple RNN cell used in the hidden component instead of an LSTM is detailed here as an alternative LMU model. It can achieve performance that is only slightly worse than that of the LMU with LSTM hidden cell, see Tables 5.5 and 5.6, but it uses substantially fewer parameters.

**Table 5.5** Validation and test mean RMSE of LMU model with simple RNN cell on both OhioT1DM editions.

| | 2018 cohort | | 2020 cohort | |
|---|---|---|---|---|
| Dataset | 30-min PH | 60-min PH | 30-min PH | 60-min PH |
| Validation | 21 | 34.37 | 21.79 | 35.62 |
| Test | 18.3 | 30.42 | 18.69 | 32.76 |

**Table 5.6** Validation and test mean RMSE of LMU model with LSTM cell on both OhioT1DM editions.

| | 2018 cohort | | 2020 cohort | |
|---|---|---|---|---|
| Dataset | 30-min PH | 60-min PH | 30-min PH | 60-min PH |
| Validation | 20.96 | 34.29 | 21.67 | 35.56 |
| Test | 18.17 | 30.33 | 18.56 | 32.57 |

Focusing on the 30-minute PH, the LMU with a simple RNN cell uses approximately 20,000 trainable parameters, as opposed to the LMU with LSTM cells, which uses 80,000 and 90,000 for the 2018 and 2020 cohorts, respectively. This difference is even further pronounced when compared to the baseline LSTM architecture, where the number of parameters ranges from 500,000 to 700,000.

The N-BEATS models proposed by Rubin-Falcone et al. [26] reportedly used 7 blocks of 300 LSTM units, while the stacked LSTM architecture by Rabby et al. [23] used 2 LSTM layers with 128 units followed by fully connected dense layers with 512, 128 neurons, and 1 output neuron. Based on a Tensorflow implementation of the stacked LSTM architecture from [23], the number of trainable parameters is approximately 300,000. An estimate was not made for the N-BEATS architecture.

A related work with a number of parameters comparable to that of LMU is a CRNN [27] with 128 LSTM units and 52,441 trainable parameters for the entire proposed architecture. Further, the GAN architecture proposed by Zhu et al. [8] utilizing 3 layers with 32 GRU units each in the generator of the GAN architecture and further three one-dimensional causal CNN layers employed in the discriminator of the GAN architecture is also comparable.

# Chapter 6

# Results and Dicussion

Table 6.1 presents the validation RMSE for the investigated models. The LMUs achieved the best results, followed by the LSTM baseline, the Informer, and the Time Series Transformer. These findings are further confirmed by Table 6.2, which provides the RMSE evaluated on the test set.

See Figure 6.1, which illustrates the overall system architecture of LMU as the final model. This model will be evaluated and compared with other related works.

**Table 6.1** Validation mean RMSE of the proposed models on both OhioT1DM editions.

| | 2018 cohort | | 2020 cohort | |
|---|---|---|---|---|
| **Model** | **30-min PH** | **60-min PH** | **30-min PH** | **60-min PH** |
| LSTM baseline | 21.55 | 35.4 | 21.98 | 36.13 |
| TS Transformer | 23.55 | 39.12 | 23.04 | 39.85 |
| Informer | 23.52 | 36.81 | 22.96 | 36.86 |
| LMU | **20.96** | **34.29** | **21.67** | **35.56** |

**Table 6.2** Test mean RMSE of the proposed models on both OhioT1DM editions.

| | 2018 cohort | | 2020 cohort | |
|---|---|---|---|---|
| **Model** | **30-min PH** | **60-min PH** | **30-min PH** | **60-min PH** |
| LSTM baseline | 18.88 | 31.32 | 18.8 | 32.96 |
| TS Transformer | 20.5 | 34.84 | 21.25 | 38.99 |
| Informer | 20.34 | 32.42 | 21.26 | 34.96 |
| LMU | **18.17** | **30.33** | **18.56** | **32.57** |

For the purpose of the final evaluation, the best LMU models were trained and evaluated 5 times. Tables 6.3 and 6.4 show the performance of the evaluated LMU models, with the mean RMSE achieved over 5 runs and the standard

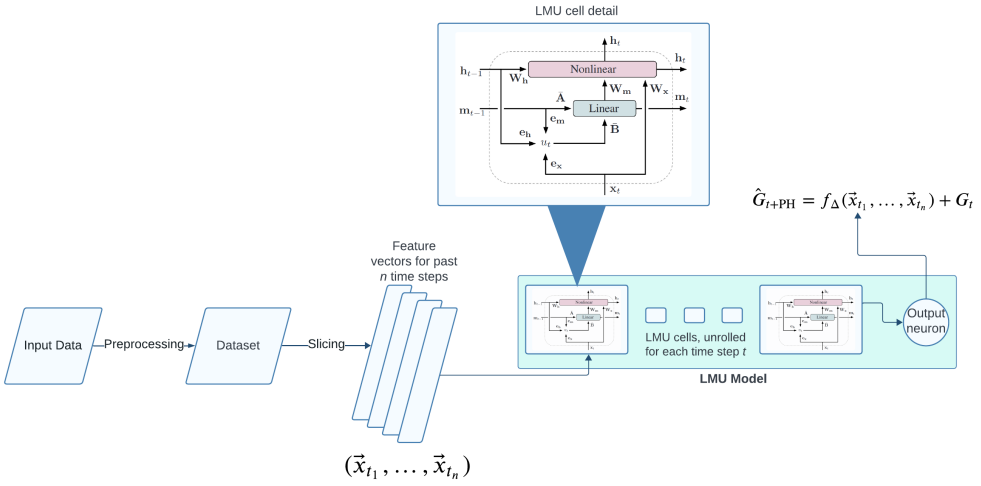



**Figure 6.1** LMU architecture (the schema for the LMU cell is sourced from Voelker et al. [37]).

deviation. Figure 6.2 (a) shows an example of predictions made by the best LMU model.

For the 2018 cohort (Table 6.3) and 30-minute PH, individual patient RMSEs range from 15.23 to 21.28 mg/dl, while for a 60-minute PH, they range from 27.15 to 33.59 mg/dl. The RMSE values are higher for patients 575 and 591, which might indicate specific challenges related to these patients' data.

In the 2020 cohort (Table 6.4), the errors are slightly higher, with mean RMSE across patients being 18.56 and 32.57 mg/dl for 30-minute and 60-minute PH, respectively. Patients 540, 567, and 584 show notably higher RMSEs for both PHs than others.

Moreover, Tables 6.7 and 6.8 show the comparison of the best LMU models with other works mentioned in the related work, evaluated on the 2018 and 2020 editions of OhioT1DM, respectively. It can be seen that the performance on the 2018 edition of the dataset is the best in the case of 30-min PH and comparable to the best in the case of the 60-min PH. One should also note that for the Stacked LSTM introduced by Rabby et al. in [23], the authors skipped testing inputs with any missing values of BG. If the same procedure is applied, the results are further improved to 18.03 mg/dl mean RMSE for 30-min PH and 30.18 mg/dl mean RMSE for 60-min PH.

The performance of LMU models on the 2020 edition of the dataset is slightly worse than the state-of-the-art model N-BEATS [26]. However, as was mentioned earlier in Section 2.2.2, they use more datasets for training which has a positive influence. Further, the GAN architecture [8] also slightly outperforms LMU models on the 2020 edition.

In addition to RMSE, the Clarke error grid [44] is constructed as a semi-quantitative tool to obtain clinical relevance. An example is provided in Fig-

**Table 6.3** Test evaluation of the best LMU model on the 2018 OhioT1DM edition.

| Patient ID | RMSE (30-min PH) | RMSE (60-min PH) |
|---|---|---|
| 559 | 17.63 ± 0.04 | 30.81 ± 0.14 |
| 563 | 17.54 ± 0.06 | 29.24 ± 0.14 |
| 570 | 15.23 ± 0.06 | 27.15 ± 0.12 |
| 575 | 21.28 ± 0.07 | 33.59 ± 0.18 |
| 588 | 16.83 ± 0.05 | 28.24 ± 0.14 |
| 591 | 20.51 ± 0.04 | 32.92 ± 0.19 |
| Mean RMSE | 18.17 ± 0.02 | 30.33 ± 0.08 |

**Table 6.4** Test evaluation of the best LMU model on the 2020 OhioT1DM edition.

| Patient ID | RMSE (30-min PH) | RMSE (60-min PH) |
|---|---|---|
| 540 | 20.50 ± 0.05 | 37.70 ± 0.14 |
| 544 | 16.51 ± 0.07 | 28.72 ± 0.17 |
| 552 | 15.72 ± 0.06 | 28.98 ± 0.11 |
| 567 | 20.41 ± 0.06 | 36.13 ± 0.15 |
| 584 | 21.92 ± 0.07 | 35.90 ± 0.16 |
| 596 | 16.32 ± 0.06 | 28.00 ± 0.13 |
| Mean RMSE | 18.56 ± 0.03 | 32.57 ± 0.09 |

ure 6.2 (b), and the values for all patients on the 2018 and 2020 editions of OhioT1DM are reported in Tables 6.5 and 6.6 respectively. The results confirm the promising performance. Clarke error grid figures for all patients and PHs are available in Appendix A.

**Table 6.5** Test Clarke error grid distribution (in %) on the 2018 OhioT1DM edition, for the best LMU model.

| | Zone for 30-min PH | | | | | Zone for 60-min PH | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Patient ID** | **A** | **B** | **C** | **D** | **E** | **A** | **B** | **C** | **D** | **E** |
| 559 | 91.7 | 7.01 | 0 | 1.29 | 0 | 75.18 | 21.13 | 0.25 | 3.4 | 0.04 |
| 563 | 93.71 | 5.82 | 0 | 0.47 | 0 | 79.29 | 19.1 | 0.08 | 1.53 | 0 |
| 570 | 97.62 | 2.2 | 0 | 0.18 | 0 | 88.46 | 11.02 | 0 | 0.52 | 0 |
| 575 | 87.42 | 9.98 | 0 | 2.6 | 0 | 67.27 | 26.08 | 0.39 | 6.23 | 0.04 |
| 588 | 93.91 | 5.94 | 0 | 0.14 | 0 | 79.69 | 19.41 | 0.04 | 0.87 | 0 |
| 591 | 82.79 | 13.93 | 0.04 | 3.24 | 0 | 62.13 | 32 | 0.29 | 5.58 | 0 |

**Table 6.6** Test Clarke error grid distribution (in %) on the 2020 OhioT1DM edition, for the best LMU model.

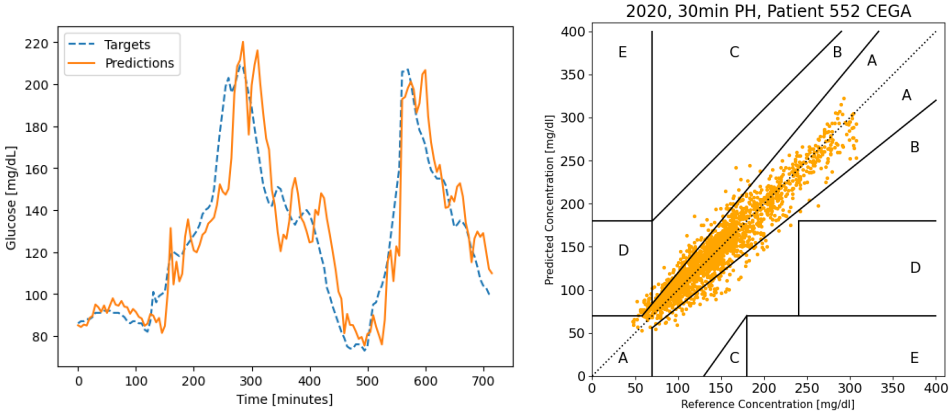
| | Zone for 30-min PH | | | | | Zone for 60-min PH | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Patient ID** | **A** | **B** | **C** | **D** | **E** | **A** | **B** | **C** | **D** | **E** |
| 540 | 85.25 | 12.43 | 0 | 2.33 | 0 | 60.06 | 34.02 | 0.1 | 5.82 | 0 |
| 544 | 92.62 | 6.9 | 0 | 0.48 | 0 | 74.02 | 24.37 | 0 | 1.61 | 0 |
| 552 | 90.48 | 8.32 | 0 | 1.2 | 0 | 66.29 | 30.23 | 0.04 | 3.4 | 0.04 |
| 567 | 86.8 | 10.94 | 0 | 2.26 | 0 | 59.57 | 31.65 | 0.3 | 8.48 | 0 |
| 584 | 87.33 | 11.64 | 0.08 | 0.95 | 0 | 69.32 | 28.16 | 0.5 | 2.02 | 0 |
| 596 | 90.82 | 7.19 | 0 | 1.99 | 0 | 73.32 | 23.71 | 0.04 | 2.94 | 0 |

**Table 6.7** Mean RMSE comparison on the 2018 OhioT1DM edition.

| **Model** | **Mean RMSE (30-min PH)** | **Mean RMSE (60-min PH)** |
|---|---|---|
| LSTM [25] | 20.1 | 33.2 |
| Dilated RNN [24] | 19.04 | not-applicable |
| Stacked LSTM [23] | 18.57 | 30.32 |
| LMU (ours) | 18.17 | 30.33 |

**Table 6.8** Mean RMSE comparison on the 2020 OhioT1DM edition.

| **Model** | **Mean RMSE (30-min PH)** | **Mean RMSE (60-min PH)** |
|---|---|---|
| GAN [8] | 18.34 | 32.21 |
| N-BEATS [26] | 18.22 | 31.66 |
| CRNN [27] | 19.79 | 33.73 |
| LMU (ours) | 18.56 | 32.57 |




**(a)** Predictions **(b)** Clarke error grid

**Figure 6.2** Example of BG predictions made by LMU (a) and the Clarke error grid plot (b) for 30-minute PH and the patient with ID 552 on 2020 OhioT1DM edition.

# Conclusion

The goal of this thesis was to research and improve ML models for BG prediction. The aim was to explore prospective ML models and training strategies, evaluate these models using the OhioT1DM dataset, and compare the results with existing studies.

The author successfully achieved these goals. Initially, state-of-the-art models were reviewed, specifically focusing on those utilizing the OhioT1DM dataset. Further, the author explored Transformer architectures and Legendre Memory Units (LMUs), with various training strategies and features. This included an ablation study that highlighted the progressive improvements in the proposed ML models.

The experiments demonstrated that LMUs outperformed both Transformers and the baseline LSTM architecture across both 30-minute and 60-minute prediction horizons. Further, they were proven to reach and, on smaller datasets (2018 edition of OhioT1DM), even outperform the state-of-the-art models. A research paper detailing the best models developed is to appear at the International Conference on Artificial Intelligence in Medicine [48].

There are likely further enhancements to be made, possibly through the integration of additional features or improved pre-processing strategies. Future research could also explore more complex architectures incorporating LMUs, such as generative adversarial networks.

# Appendix A
# Appendix

Clarke error grid [44] figures for all patients and PHs, evaluated on the best LMU models.

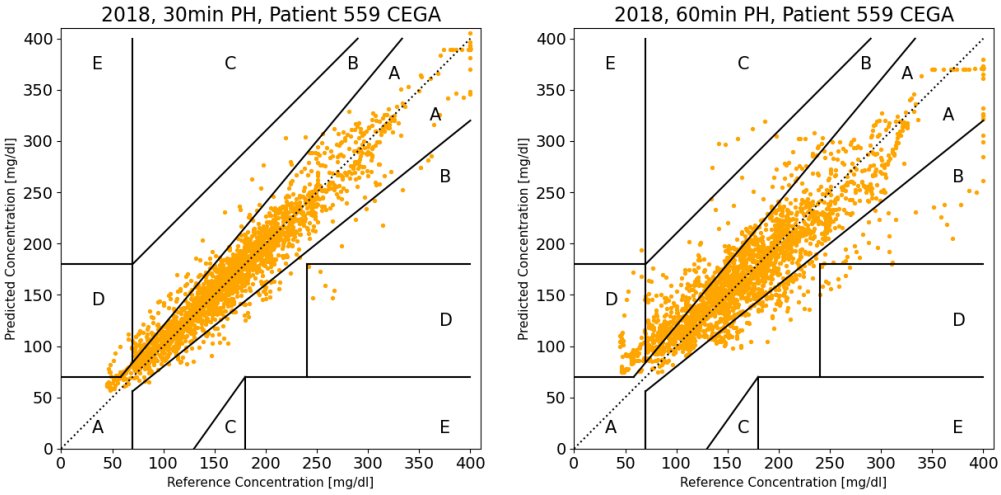



**Figure A.1** CEGA for patient 559.

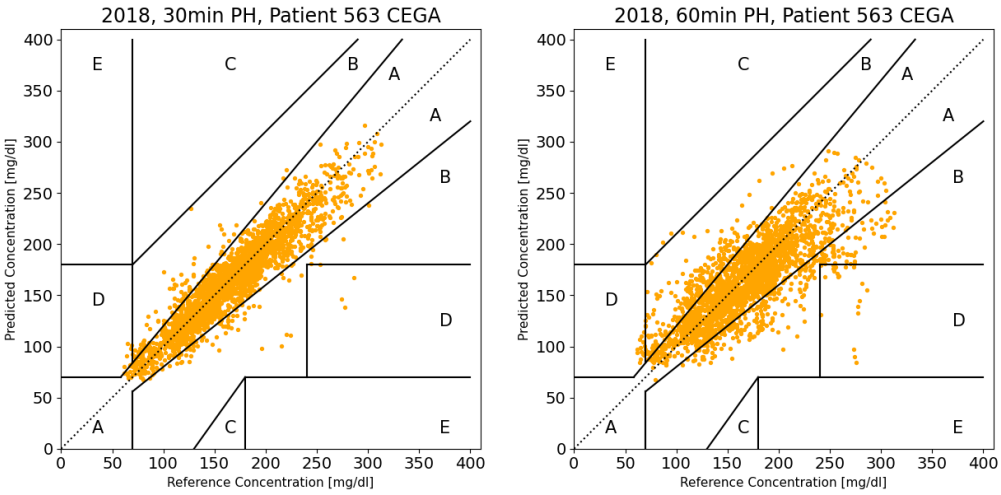



**Figure A.2** CEGA for patient 563.

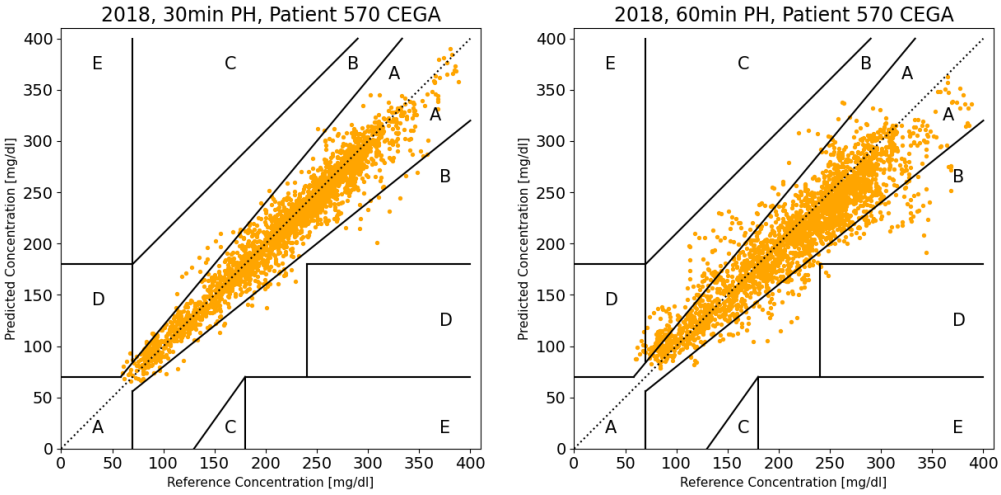



**Figure A.3** CEGA for patient 570.

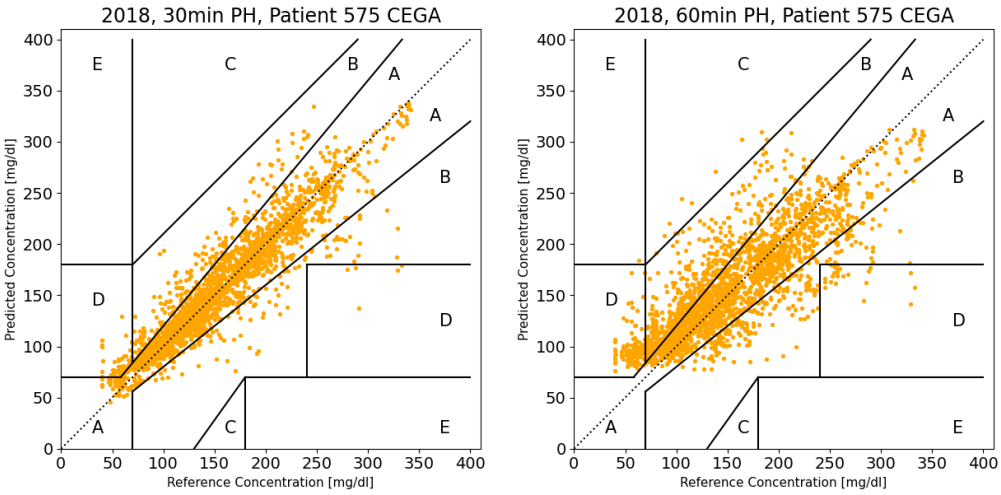



**Figure A.4** CEGA for patient 575.

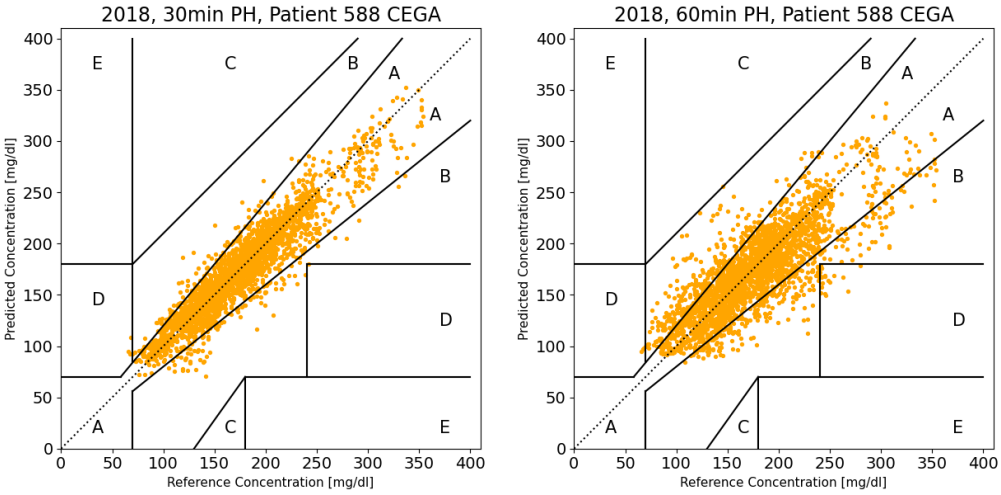



**Figure A.5** CEGA for patient 588.

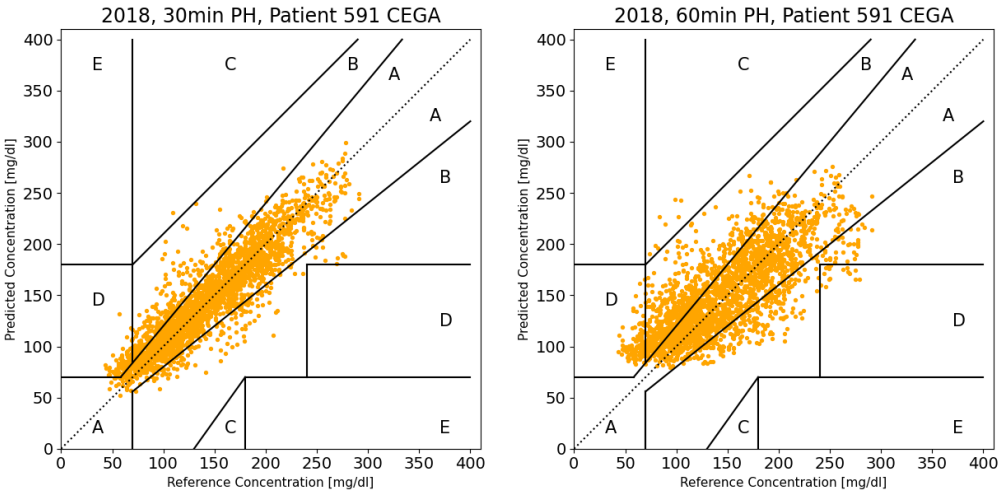



**Figure A.6** CEGA for patient 591.

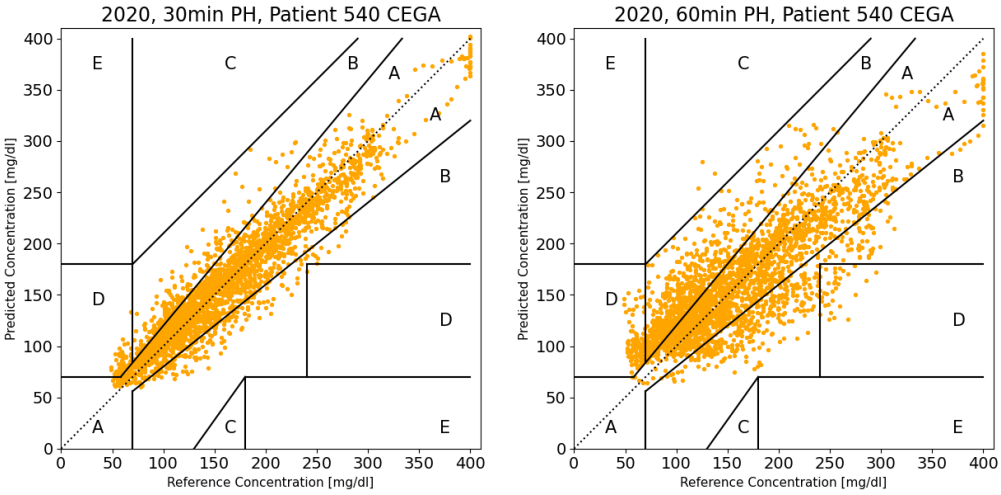



**Figure A.7** CEGA for patient 540.

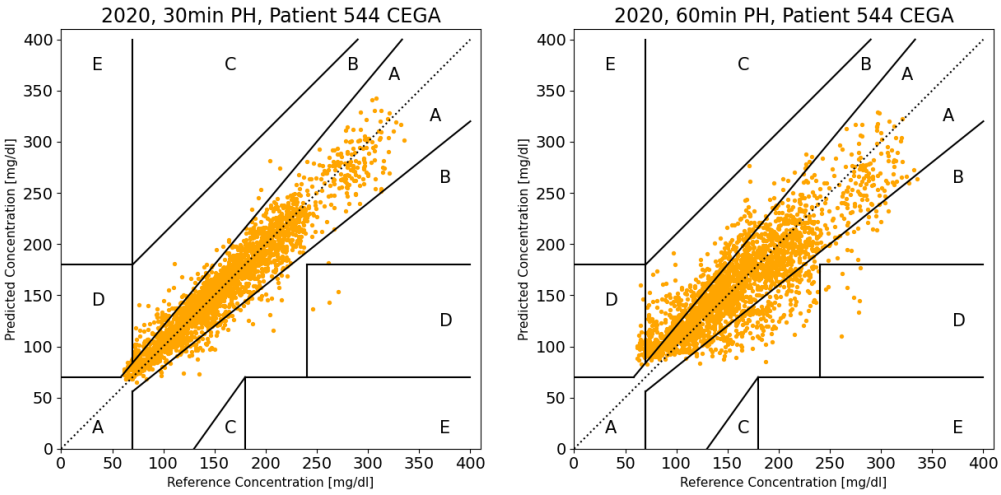



**Figure A.8** CEGA for patient 544.

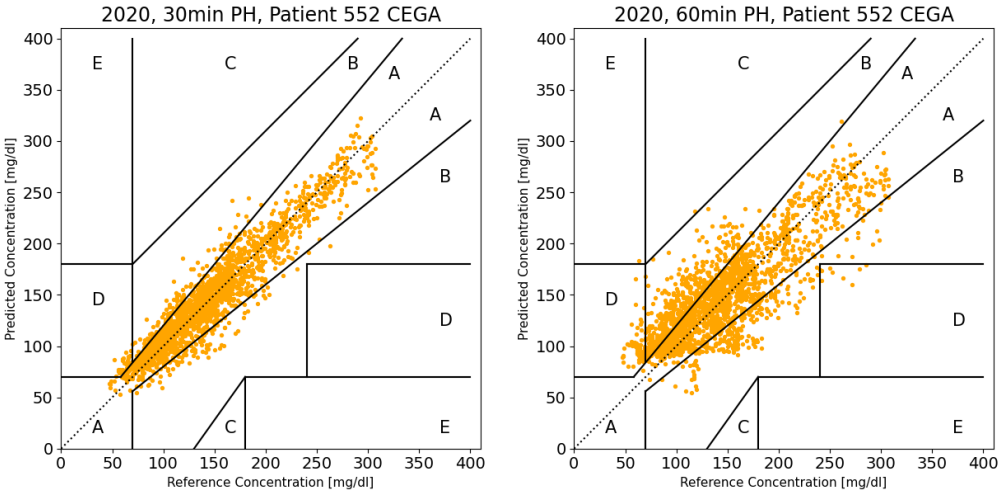



**Figure A.9** CEGA for patient 552.

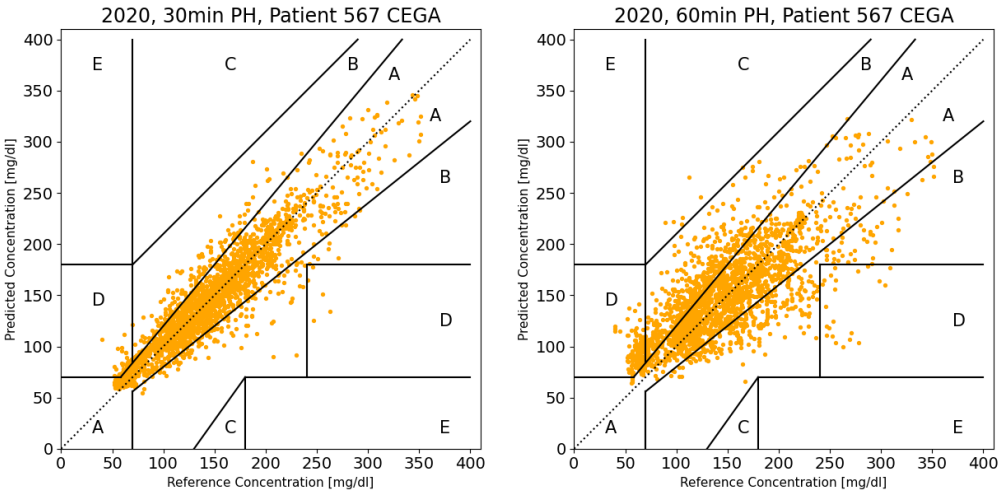



**Figure A.10** CEGA for patient 567.

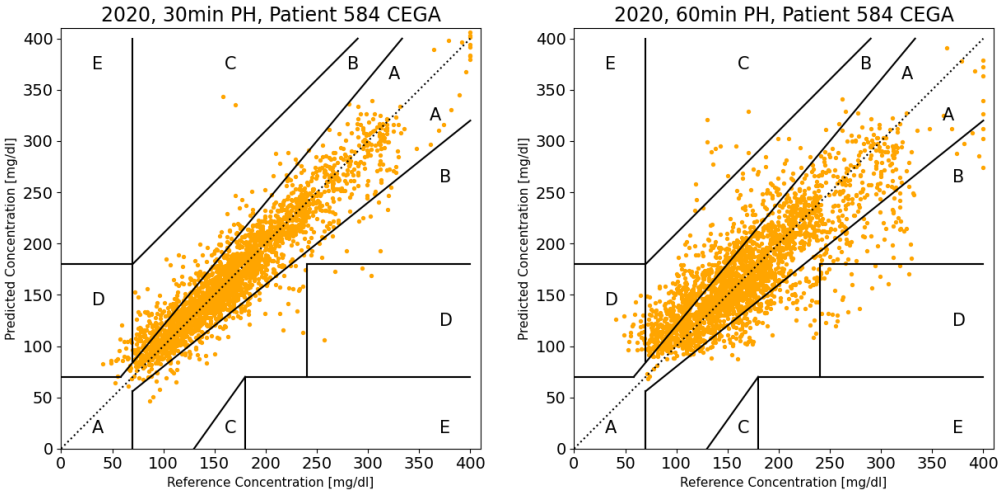



**Figure A.11** CEGA for patient 584.

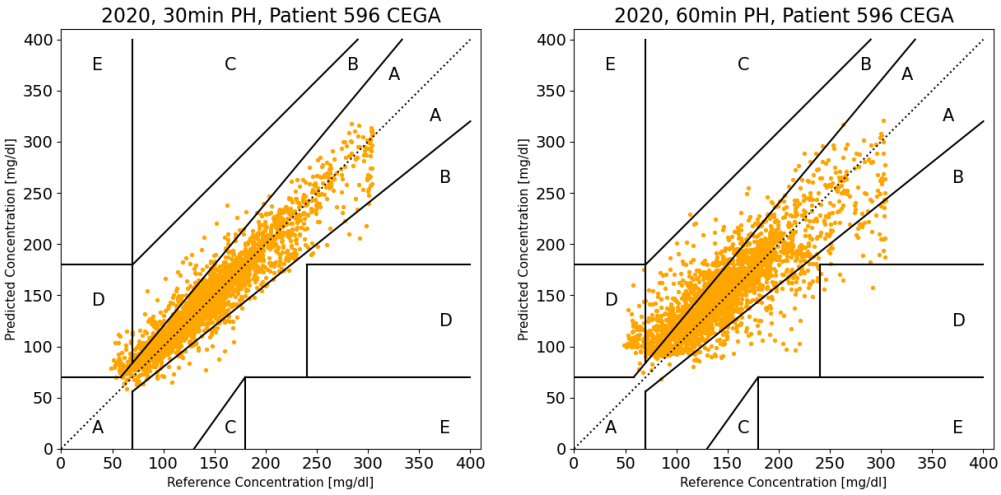



**Figure A.12** CEGA for patient 596.

# Bibliography


1. MARLING, CIndy; BUNESCU, Razvan. The OhioT1DM Dataset for Blood Glucose Level Prediction. In: *Proceedings of the 5th International Workshop on Knowledge Discovery in Healthcare Data*. CEUR Workshop Proceedings, 2020. Available at `http://smarthealth.cs.ohio.edu/bglp/OhioT1DM-dataset-paper.pdf`.
2. *Glucose Tolerance test* [online]. [visited on 2024-04-23]. Available from: `www.medlineplus.gov/ency/article/003466.htm`.
3. MATHEW, P.; THOPPIL, D. *Hypoglycemia* [StatPearls [Internet]]. Updated 2022 Dec 26. Treasure Island (FL): StatPearls Publishing, 2024. Available from: `https://www.ncbi.nlm.nih.gov/books/NBK534841/`.
4. MOURI, M.I.; BADIREDDY, M. *Hyperglycemia* [StatPearls [Internet]]. Updated 2023 Apr 24. Treasure Island (FL): StatPearls Publishing, 2024. Available from: `https://www.ncbi.nlm.nih.gov/books/NBK430900/`.
5. FOSTER-POWELL, K; MILLER, J B. International tables of glycemic index. *The American Journal of Clinical Nutrition*. 1995, vol. 62, no. 4, 871S–890S. ISSN 0002-9165. Available from DOI: `10.1093/ajcn/62.4.871S`.
6. DAVIDSON, Paul C; HEBBLEWHITE, Harry R; STEED, Robert D; BODE, Bruce W. Analysis of guidelines for basal-bolus insulin dosing: basal insulin, correction factor, and carbohydrate-to-insulin ratio. *Endocrine Practice*. 2008, vol. 14, no. 9, pp. 1095–1101. Available also from: `https://doi.org/10.4158/EP.14.9.1095`.
7. KROLL, Martin H. Biological variation of glucose and insulin includes a deterministic chaotic component. *Biosystems*. 1999, vol. 50, no. 3, pp. 189–201. ISSN 0303-2647. Available from DOI: `https://doi.org/10.1016/S0303-2647(99)00007-6`.

8. ZHU, Taiyu; YAO, Xi; LI, Kezhi; HERRERO, Pau; GEORGIOU, Pantelis. Blood glucose prediction for type 1 diabetes using generative adversarial networks. In: *CEUR Workshop Proceedings*. 2020, vol. 2675, pp. 90–94.

9. NEINSTEIN, Aaron; WONG, Jenise; LOOK, Howard; ARBITER, Brandon; QUIRK, Kent; MCCANNE, Steve; SUN, Yao; BLUM, Michael; ADI, Saleh. A case study in open source innovation: developing the Tidepool Platform for interoperability in type 1 diabetes management. *Journal of the American Medical Informatics Association*. 2016, vol. 23, no. 2, pp. 324–332.

10. *OpenAPS Data Commons* [online]. [visited on 2024-04-23]. Available from: `https://OpenAPS.org/data-commons`.

11. PRIOLEAU, Temiloluwa; BARTOLOME, Abigail; COMI, Richard; STANGER, Catherine. DiaTrend: A dataset from advanced diabetes technology to enable development of novel analytic solutions. *Scientific Data*. 2023, vol. 10, no. 1, p. 556.

12. *Kaggle Type 1 Diabetes dataset* [online]. [visited on 2024-04-23]. Available from: `www.kaggle.com/datasets/lacofloris/type-1-diabetes-blood-glucose-prediction`.

13. MAN, Chiara Dalla; MICHELETTO, Francesco; LV, Dayu; BRETON, Marc; KOVATCHEV, Boris; COBELLI, Claudio. The UVA/PADOVA type 1 diabetes simulator: new features. *Journal of diabetes science and technology*. 2014, vol. 8, no. 1, pp. 26–34.

14. *Medtronic 530G picture* [online]. [visited on 2024-04-23]. Available from: `https://www.medtronicdiabetes.com/download-library/minimed-530g`.

15. *Medtronic Enlite sensor picture* [online]. [visited on 2024-04-23]. Available from: `https://eshop.medtronic-diabetes.co/en/cgmsupplies/sensors/EnliteSensor`.

16. BASU, Ananda; DUBE, Simmi; SLAMA, Michael; ERRAZURIZ, Isabel; AMEZCUA, Jose Carlos; KUDVA, Yogish C.; PEYSER, Thomas; CARTER, Rickey E.; COBELLI, Claudio; BASU, Rita. Time Lag of Glucose From Intravascular to Interstitial Compartment in Humans. *Diabetes*. 2013, vol. 62, no. 12, pp. 4083–4087. ISSN 0012-1797. Available from DOI: `10.2337/db13-1132`.

17. *Support materials for MiniMed® 530G with Enlite®* [online]. [visited on 2024-04-23]. Available from: `https://www.medtronicdiabetes.com/sites/default/files/library/support/Getting%20Started%20with%20CGM%20for%20the%20Minimed%20530G%20wtih%20Enlite.pdf`.

18. BREMER, Troy; GOUGH, David A. Is blood glucose predictable from previous values? A solicitation for data. *Diabetes*. 1999, vol. 48, no. 3, pp. 445–451. Available also from: `https://diabetesjournals.org/diabetes/article/48/3/445/12150/Is-blood-glucose-predictable-from-previous-values`.

19. WOLDAREGAY, Ashenafi Zebene; ÅRSAND, Eirik; WALDERHAUG, Ståle; ALBERS, David; MAMYKINA, Lena; BOTSIS, Taxiarchis; HARTVIGSEN, Gunnar. Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes. *Artificial Intelligence in Medicine*. 2019, vol. 98, pp. 109–134. ISSN 0933-3657. Available from DOI: `https://doi.org/10.1016/j.artmed.2019.07.007`.

20. GEORGA, Eleni I.; PROTOPAPPAS, Vasilios C.; ARDIGÒ, Diego; MARINA, Michela; ZAVARONI, Ivana; POLYZOS, Demosthenes; FOTIADIS, Dimitrios I. Multivariate Prediction of Subcutaneous Glucose Concentration in Type 1 Diabetes Patients Based on Support Vector Regression. *IEEE Journal of Biomedical and Health Informatics*. 2013, vol. 17, no. 1, pp. 71–81. Available from DOI: `10.1109/TITB.2012.2219876`.

21. YANG, Jun; LI, Lei; SHI, Yimeng; XIE, Xiaolei. An ARIMA Model With Adaptive Orders for Predicting Blood Glucose Concentrations and Hypoglycemia. *IEEE Journal of Biomedical and Health Informatics*. 2019, vol. 23, no. 3, pp. 1251–1260. Available from DOI: `10.1109/JBHI.2018.2840690`.

22. GEORGA, Eleni I.; PROTOPAPPAS, Vasilios C.; ARDIGÒ, Diego; MARINA, Michela; ZAVARONI, Ivana; POLYZOS, Demosthenes; FOTIADIS, Dimitrios I. Multivariate Prediction of Subcutaneous Glucose Concentration in Type 1 Diabetes Patients Based on Support Vector Regression. *IEEE Journal of Biomedical and Health Informatics*. 2013, vol. 17, no. 1, pp. 71–81. Available from DOI: `10.1109/TITB.2012.2219876`.

23. RABBY, Md Fazle; TU, Yazhou; HOSSEN, Md Imran; LEE, Insup; MAIDA, Anthony S; HEI, Xiali. Stacked LSTM based deep recurrent neural network with kalman smoothing for blood glucose prediction. *BMC Medical Informatics and Decision Making*. 2021, vol. 21, pp. 1–15.

24. CHEN, Jianwei; LI, Kezhi; HERRERO, Pau; ZHU, Taiyu; GEORGIOU, Pantelis. Dilated Recurrent Neural Network for Short-time Prediction of Glucose Concentration. In: *KHD@ IJCAI*. 2018, pp. 69–73.

25. MARTINSSON, John; SCHLIEP, Alexander; ELIASSON, Björn; MEIJNER, Christian; PERSSON, Simon; MOGREN, Olof. Automatic blood glucose prediction with confidence using recurrent neural networks. In: *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data, KDH@IJCAI-ECAI 2018*. 2018, pp. 64–68.

26. RUBIN-FALCONE, Harry; FOX, Ian; WIENS, Jenna. Deep Residual Time-Series Forecasting: Application to Blood Glucose Prediction. *KDH@ ECAI*. 2020, vol. 20, pp. 105–109.

27. DANIELS, John; HERRERO, Pau; GEORGIOU, Pantelis. Personalised Glucose Prediction via Deep Multitask Networks. *KDH@ ECAI*. 2020, vol. 20, pp. 110–114.

28. HOVORKA, Roman; CANONICO, Valentina; CHASSIN, Ludovic J; HAUETER, Ulrich; MASSI-BENEDETTI, Massimo; FEDERICI, Marco Orsini; PIEBER, Thomas R; SCHALLER, Helga C; SCHAUPP, Lukas; VERING, Thomas; WILINSKA, Malgorzata E. Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes. *Physiological Measurement*. 2004, vol. 25, no. 4, p. 905.

29. PRICE, Thony. *Working repository for Master Thesis* [`https://github.com/ThonyPrice/Master_Thesis/`]. GitHub, 2019.

30. HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long Short-Term Memory. *Neural Computation*. 1997, vol. 9, no. 8, pp. 1735–1780. Available from DOI: `10.1162/neco.1997.9.8.1735`.

31. GERS, Felix A; SCHMIDHUBER, Jürgen; CUMMINS, Fred. Learning to forget: Continual prediction with LSTM. *Neural computation*. 2000, vol. 12, no. 10, pp. 2451–2471.

32. OLAH, Christopher. *Understanding LSTM Networks* [online]. [visited on 2024-04-23]. Available from: `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

33. VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N; KAISER, Łukasz; POLOSUKHIN, Illia. Attention is all you need. *Advances in neural information processing systems*. 2017, vol. 30.

34. *Time Series Transformer implementation* [online]. [visited on 2024-04-23]. Available from: `https://huggingface.co/docs/transformers/en/model_doc/time_series_transformer`.

35. ZHOU, Haoyi; ZHANG, Shanghang; PENG, Jieqi; ZHANG, Shuai; LI, Jianxin; XIONG, Hui; ZHANG, Wancai. Informer: Beyond efficient transformer for long sequence time-series forecasting. In: *Proceedings of the AAAI conference on artificial intelligence*. 2021, vol. 35, pp. 11106–11115. No. 12.

36. *Informer implementation* [online]. [visited on 2024-04-23]. Available from: `https://huggingface.co/docs/transformers/en/model_doc/informer`.

37. VOELKER, Aaron; KAJIĆ, Ivana; ELIASMITH, Chris. Legendre Memory Units: Continuous-Time Representation in Recurrent Neural Networks. *Advances in Neural Information Processing Systems*. 2019, vol. 32.

38. *LMU implementation documentation* [online]. [visited on 2024-04-23]. Available from: `https://www.nengo.ai/keras-lmu/api-reference.html`.

39. PADÉ, Henri. Sur la représentation approchée d'une fonction par des fractions rationnelles. In: *Annales scientifiques de l'Ecole normale supérieure*. 1892, vol. 9, pp. 3–93.

40. VOELKER, Aaron Russell. Dynamical systems in spiking neuromorphic hardware. 2019.

41. LEGENDRE, Adrien-Marie. Recherches sur l'attraction des sphéroïdes homogènes. *Mémoires de Mathématiques et de Physique, présentés à l'Académie Royale des Sciences*. 1782, pp. 411–435.

42. RODRIGUES, Olinde. *De l'attraction des sphéroïdes*. 1816. PhD thesis. University of Paris. Correspondence sur l'École Impériale Polytechnique.

43. CHANDAR, Sarath; SANKAR, Chinnadhurai; VORONTSOV, Eugene; KAHOU, Samira Ebrahimi; BENGIO, Yoshua. Towards Non-Saturating Recurrent Units for Modelling Long-Term Dependencies. *arXiv preprint arXiv:1902.06704*. 2019.

44. CLARKE, William L; COX, Daniel; GONDER-FREDERICK, Linda A; CARTER, William; POHL, Stephen L. Evaluating Clinical Accuracy of Systems for Self-Monitoring of Blood Glucose. *Diabetes Care*. 1987, vol. 10, no. 5, pp. 622–628. ISSN 0149-5992. Available from DOI: `10.2337/diacare.10.5.622`.

45. KINGMA, Diederik P; BA, Jimmy. Adam: A method for stochastic optimization. In: *International Conference on Learning Representations*. 2015, pp. 1–15.

46. LI, Lisha; JAMIESON, Kevin; DESALVO, Giulia; ROSTAMIZADEH, Afshin; TALWALKAR, Ameet. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. *Journal of Machine Learning Research*. 2018, vol. 18, no. 185, pp. 1–52. Available also from: `http://jmlr.org/papers/v18/16-558.html`.

47. CHEN, Xiangning; LIANG, Chen; HUANG, Da; REAL, Esteban; WANG, Kaiyuan; PHAM, Hieu; DONG, Xuanyi; LUONG, Thang; HSIEH, Cho-Jui; LU, Yifeng, et al. Symbolic discovery of optimization algorithms. *Advances in Neural Information Processing Systems*. 2024, vol. 36.

48. FLORIŠ, Ladislav; VAŠATA, Daniel. Predicting Blood Glucose Levels with LMU Recurrent Neural Networks: A Novel Computational Model. In: *Artificial Intelligence in Medicine*. 2024. To appear.

# Contents of the attached media

- `code`
  - `ablation` ..... jupyter notebooks containing the ablation study
  - `LMUs` jupyter notebooks with the implementation and training of LMU models
  - `LSTMs` .... jupyter notebooks with the implementation and training of LSTM models
  - `transformers` ..... jupyter notebooks with the implementation and training of transformer models
  - `misc` . various jupyter notebooks for analysis, evaluation, generation of figures and additional model implementations
  - `hyperparameters_tune.py` ..... script for hyper-parameter tuning of LSTM and LMU architectures
  - `tune_transformers.py` ..... script for hyper-parameter tuning of transformer architectures
- `core` ..... python files with common logic for model training and dataset pre-processing
- `data` .. should contain the OhioT1DM dataset files, left empty due to the dataset licensing
- `plots` ..... contains figures used in the thesis
- `tests` ..... python tests
- `requirements.txt` ..... list of python dependencies
- `README.md` ..... introduction of the implementation and instructions for running it
- `thesis.pdf` ..... pdf of the thesis
- `thesis_src.zip` ..... LaTeX thesis source code