Third Year Project Report - Blood Glucose Forecasting using Machine Learning

A Report submitted to The University of Manchester for the degree of

A Bachelor of Science in Computer Science

in the Faculty of Science and Engineering

Year of submission

2023

Student ID

10444901

School of Engineering

Contents

Word count: 10000

Acknowledgements

I would like to thank my tutor Paul Nutter for all of his knowledge and advice throughout the duration of this project.

Abstract

This is abstract text.

1 Introduction

1.1 Background and motivation

Talk about diabetes generally, can take from introduction of lit review

Type 1 Diabetes Mellitus (T1DM) is a life-long autoimmune disease that typically manifests itself in children between 10 and 14 years of age. That is not to say, however, that a diagnosis of T1DM is restricted to those individuals within this age range, and whilst it is rare for diagnosis to be made in individuals over 40, it is possible for a diagnosis of T1DM to occur at any age. T1DM is characterised by the destruction of the beta-cells within the pancreas which are responsible for producing insulin [1], a hormone used by the body to help blood glucose to be absorbed by cells to be used for energy [2], and to store blood glucose for later use. This loss of insulin producing beta cells means that there is an inability for T1DM patients to regulate their own blood glucose levels, which if left untreated can lead to hyperglycemia (blood glucose levels being too high). As such, T1DM patients are reliant on the subcutaneous injection of synthetic insulin to control the glycemic levels in their body and to maintain a healthy blood glucose level. Diagnosis of the condition is based on hyperglycemia, with a fasting blood glucose concentration above 7.0 mmol/L being characterised as enough to diagnose. Other symptoms commonly seen before diagnosis are going to the toilet a lot, constantly being thirsty, and weight loss [3].

In order to calculate how much external insulin to inject, and thus avoid hyperglycemia and hypoglycemia (blood glucose levels being too low), T1DM patients must keep track of how much glucose they ingest and dose themselves with a corresponding volume of insulin. This counting and dosing process is not an exact science, and being inaccurate can lead to potentially life-threatening situations for patients. As such, technology has been introduced to make the process easier. Continuous Glucose Monitoring (CGM) devices [4] provide real-time information on someone's current Blood Glucose (BG) levels, whilst insulin pumps exist that are capable of slowly providing insulin over a long period of time, rather than having a single large dose given by injection pen. This new technology has greatly improved the lives of many patients, and there is hope that new technology can continue to do the same. One of the main issues with current technology is that whilst it is able to immediately detect issues with current BG levels, there is still a response time between corrective actions being taken and their impacts on BG levels. This response time has lead to much research being done into predictive algorithms that could inform a patient ahead of time of any issues that may occur, allowing them to react with ample time and avoid any high/low blood glucose levels before they even manifest. One such branch of these predictive algorithms is machine learning algorithms and an exploration of this topic will form the basis of this project report.

1.2 Aims and objectives

This project aims to investigate and implement machine learning methods to perform blood glucose prediction, then compare and contrast the performance of these different methods.

To guide the development of the project, the following objectives have been identified:

- Obtain, understand and configure a suitable Continuous Glucose Monitoring (CGM) dataset.

- Perform a literature review on existing approaches taken to perform blood glucose prediction.

- Implement a selection of the researched approaches.

- Experiment with hyperparameters on different approaches to try and achieve optimal accuracy for the approach as stated by scientific metrics.

- Experiment with the different approaches over varying prediction horizons.

- Produce visualisations to view the performance of the different prediction techniques over differing time horizons.

- Analyse and compare the effectiveness of the different approaches blood glucose prediction.

- Discuss the work that would need to be done to make such prediction technologies available and useful to real patients.

1.3 Report structure

In this report, there is first a short literature review to gain insight into existing machine learning techniques for BG prediction as well as their respective performances. A few approaches are then selected from those researched, and the development and optimisation process of the different models is discussed and documented. Once finished, the models are experimented with and the data produced is visualised in the report. Finally, conclusions and insights into future work are drawn from the visualisations presented.

2 Literature review

2.1 Introduction

It is important to note that when measuring BG levels, there are two prevalent units - mmol/L which is used in the UK, and mg/dL which is used outside of the UK. The conversion ratio between

the two is as follows:

$$18mg/dL : 1mmol/L$$

According to the World Health Organisation (WHO), blood glucose levels should be between 3.9 mmol/L and 5.6mmol/L [5] for ordinary people and between 5 mmol/L and 9 mmol/L for diabetic patients. As a result of this, there is a constant need for diabetics to manage their blood glucose levels, with the aim of spending as much time within this range as possible and avoiding hyperglycemia and hypoglycemia. Both hyperglycemia and hypoglycemia are serious conditions that can lead to life threatening situations for patients; making precision when managing blood glucose levels a matter of life and death. Patients who spend extended periods of time in a hyperglycemic state can suffer from kidney damage, nerve damage (neuropathy), damage to the blood vessels of the retina leading to blindness, and complications with bones and joints. In extreme cases of hyperglycemia, diabetic ketoacidosis (DKA) can occur, which, if left untreated, can lead to a life-threatening diabetic coma [6] Meanwhile, hypoglycemia affords a range of different issues, ranging from slurred speech and numbness to seizures, coma and death in extreme cases if left untreated[7].

With an estimated 422 million people worldwide suffering from diabetes [8], making it the 3rd most common chronic illness in the world, it is clear that there is a need, and a great benefit to be gained, from using technology to help patients more accurately manage their blood glucose levels. Unfortunately, regulating blood glucose levels is not an exact science, and there are a myriad of different factors that can impact a patient's blood glucose levels [9]. The predominant factor is carbohydrates ingested by the patient. Carbohydrates are broken down by the amylase enzyme into glucose which is transferred into the blood stream to be absorbed by cells; clearly having a direct impact on the levels of glucose in the blood. Other major factors include any physical activity done by the patient, and any external insulin taken by the patient. Whilst these factors are more easily measurable for patients, and thus can be adapted to and controlled, other factors such as stress, illness and injury, and hormonal changes can also have a substantial impact on regulating blood glucose levels, whilst being much more difficult to manage. This makes the task of predicting how blood glucose levels will change and respond accordingly difficult for T1DM patients and machines alike - one small change can lead to large variations in behaviour.

The end vision of diabetes related technology is known as the Artificial Pancreas (AP) and consists of a fully closed-loop system where a T1DM patient would have to do no management of their blood glucose level at all; the AP would handle everything. Some AP systems currently exist, both as home-brew systems and regulated healthcare systems [10] but these systems operate in a hybrid manner, and still require some level of input from the user. Unfortunately, due to the problems in variation and some other issues we will cover below, the creation of a completely hands-free AP system still seems to be far in the future. However, that is not to say that technology has had no impact on the lives of T1DM patients, as there have been many advances which have made noticeable impacts on improving the quality of life of patients. Introduced in 1985 [11], insulin pens are devices that allow patients to inject insulin directly into their body. Coupled together with small,

easy to use glucometers which were introduced in the 1980s, patients were able to frequently monitor their blood glucose levels, and respond accordingly by either using their insulin pens, or ingesting more glucose, to control their blood glucose levels.

Fig. 1. An image showing a modern insulin pen and glucometer [12]

This method of self-monitoring of blood glucose (SMBG) has significant drawbacks as blood is sampled intermittently, providing only short insights of glucose concentrations, whilst ignoring ongoing glucose fluctuations. With the idea of being able to monitor blood glucose at all times, Continuous Glucose Monitoring (CGM) devices were introduced that measure glucose concentrations at intervals (usually between 1-5 min), and then transmits/stores those values. Alongside the invention of the insulin pump, a device that allows programmable infusions of insulin over long, or short, periods of time, diabetic patients were given a much more reliable way of constantly monitoring and regulating their blood glucose levels.
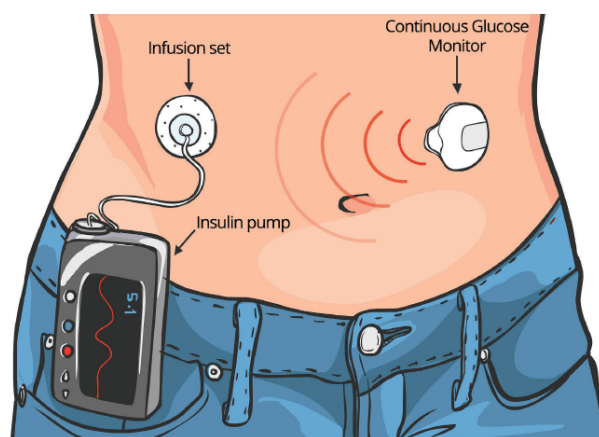

Fig. 2. A diagram showing the interaction between a CGM and insulin pump[13]

The creation of CGM devices also lead to a new capability - the ability to create blood glucose datasets by accessing the data stored in the CGM; datasets which are needed to train accurate Machine Learning (ML) models. It is clear that the capability to accurately predict when one might suffer a hyperglycemic, or hypoglycemic, episode would be invaluable to a T1DM patient, as it would

allow them to behave proactively to prevent the episode from ever occurring, rather than the current reactionary method, which requires a hypo/hyperglycemic episode to be taking place before we can detect its presence. We name the amount of time into the future which we forecast the Prediction Horizon (PH). Current prediction horizons available to T1DM patients are very short, and thus don't give ample time for patients to react to predicted issues. Current CGM devices are capable of crude alarm systems that can inform patients if their BG levels are steeply dropping or rising, but once again this is just a reactionary measure. As such, much work has been done in applying ML techniques to improve the length of the PH to a length of hours instead of minutes, but as of yet no concrete solution to the problem has been found. This is due to a variety of reasons, such as the variability of a patient's BG level, due to day-to-day uncontrollable factors as mentioned earlier, as well as some other issues that will be discussed later when different approaches that have been experimented with to solve this very problem are explored.

2.2 Detail

2.3 Machine Learning

Add section about papers / machine learning in general. Will consider whether to put ML here or later on.

Machine learning is a field of Artificial Intelligence that focuses on learning through observing patterns in large quantities of past data, and iteratively improving accuracy at a given task [14]. In the context of T1DM, the large quantities of past data refer to the patient's blood glucose levels over a period of months (as well as other points of interest such as exercise undertaken / carbohydrate intake / insulin intake), whilst the task to improve at is predicting the BG levels of the patient for a specified PH. There is interest in the use of Artificial Neural Networks (ANN), a kind of Machine learning model that is able to learn very complex relationships between data [15]. ANNs are based off of the functioning of the human brain, and are capable of modifying their structure to suit the task at hand. They consist of a number of nodes and a corresponding number of connections which link one node to another, where each connection has a weight which signals how strong the connection between the two nodes is. An ANN can consist of many layers with the first and last layers being known as the input and output layers respectively. Any layers inbetweeen the two are called hidden layers. Each node has an input into it, and in the final layer of the network, an output is given. At each node, every input is multiplied to its corresponding weight, and an activation function is applied to sum of these values to generate the output value to be passed to the next layer of the network. An activation function is a function that transforms the output in some way - for example the binary step activation function checks if the value is above a given threshold. If so, the node outputs 1, otherwise 0.
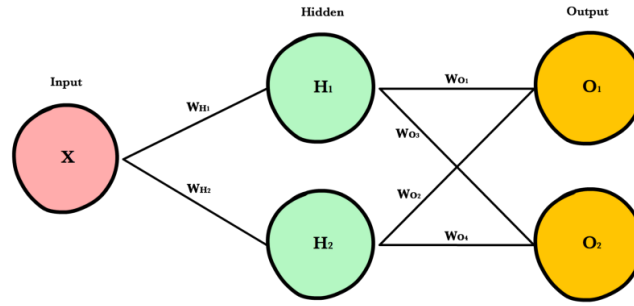
Fig. 3. A diagram showing the nodes, connections and weights of a simple ANN [16]

In order for the NN to learn the task at hand, we define an error function between the actual output of a given input and the output generated by our network. This error function can be specified to suit the model as needed and it is common, in the case of blood glucose forecasting, to see an error function comparing the absolute difference between predicted blood glucose levels, and real recorded blood glucose levels. Through an iterative algorithm such as gradient descent [17], whereby we calculate the derivative of the error function and attempt to iteratively adjust the network weights to reach the minimum value of this function, the ANN can learn very complex relationships between the data input into it and the expected outputs. When updating the network's weights during gradient descent, a parameter known as the learning rate controls how much or how little we adjust the weights by at each step of the algorithm.
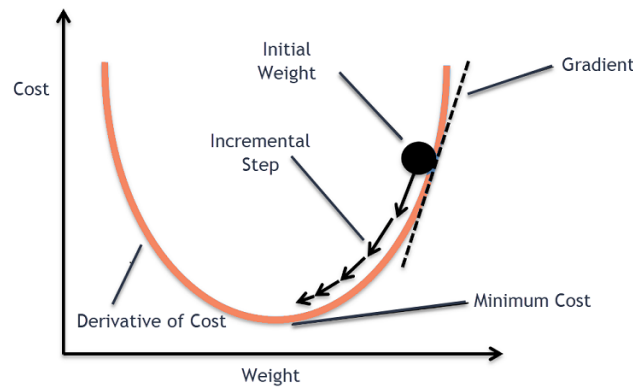

Fig. 4. A visualisation of the gradient descent process[18]

Much work has been done researching the applications of different kinds of ANN to blood glucose forecasting, as it was believed that not only would ANNs be able to provide very accurate predictions on future BG levels, but also would be able to solve the problem of inter-subject variability - by training a new network on each individual patient's BG data.

As will be seen below, when training neural networks, there is much debate amongst researchers about what to include and what not to include within the input parameters to the network. Some believe that just the CGM data is sufficient, claiming that the inclusion of other measures, such as carbohydrate intake, would simply increase variance and decrease model performance. This is

sharply contrasted by researchers who have experimented with both complex and simple physiological models to try and model the effect that factors such as exercise and insulin intake will have on future glucose levels. There is much debate about the correlation between past glucose levels and future levels, with some using only the current blood glucose levels as an input to the model, and others using many past readings to inform future predictions. Whilst there is much diversity amongst the methods employed, unfortunately, the overarching theme from most ANN approaches is that they can be very accurate at shorter PH (30 minutes), but that accuracy quickly drops off at larger PH (6 hours). Moreover, the vast majority of work done has been in silico, and such prediction methods are yet to be rigorously tested on patients in their day-to-day lives.

2.4 Simple ANN Approaches

One such example of the application of NNs to the task of blood glucose prediction was by Pérez-Gandía et al.[19], who attempted to use a NN trained using only CGM data. The model considered the preceding 20 minutes of CGM data and attempted to predict BG levels at 3 different PHs of 15, 30 and 45 minutes. The accuracy of the model was measured using a number of different metrics. One such metric is the Root Mean Squared Error (RMSE) - a widely used metric for BG prediction models that takes the square root of the sum of the squares of all prediction errors.

$$RMSE = \sqrt{\frac{\Sigma_{i=1}^{N}\Big(Predicted_i - Actual_i\Big)^2}{N}}$$

Another metric used was the Prediction Delay (PD) - a metric that takes into account the delay between when certain features of the CGM data were seen in the original data and the predicted data (for instance the time gap between a peak appearing in the original and predicted data). When making a prediction, two different approaches were compared - a standard ANN model that would make one prediction across the PH, and an AutoRegressive Model (ARM). In an ARM, the model makes the prediction one step at a time and uses a recursive algorithm to allow each prediction to inform subsequent predictions, trying to capture the relationships between previous and future BG levels more accurately. It was found that across the 15, 30 and 45 minute PHs their ANN model achieved a RMSE of 10, 18, 27, mg/dL respectively. To analyse these results, we introduce the notion of an 'accurate prediction' which, by using the accuracy of glucose sensors [20], we define to be a prediction within 15 mg/dL (0.85 mmol/L) of the actual value. For the short PH of 15 min, these results are quite promising and accurate, but this accuracy very quickly falls off. For upward trends (when the patient enters hyperglycemia rapidly) in the data, there was a PD of 4, 9 and 14 minutes respectively, and for downward trends (when the patient enters hypoglycemia rapidly), a PD of 5, 15 and 26 minutes was seen. Whislt the PD is worse when detecting downward trends, these results are still promising as at the 45 min PH, a patient would still have a 31 min period to react to an upward trend, which would give them ample time to make corrective action.

This use of a basic NN highlights some of the main problems with this approach when applied to blood glucose prediction. First, the accuracy falls of very quickly, with 2.7 times worse performance over a short difference in time of 30 minutes, and second is that the model is much quicker at recognising upward than downward trends. This is a major issue, as downward trends lead to hypoglycemic episodes, which can be far more dangerous in the short term than hyperglycemic episodes. In the case of this model, the 26 min PD at a 45 min PH would give some warning, but still wouldn't give a patient ample time to take corrective action. To try and combat the first of these problems, Alfian et al. [21] included time-domain attributes along with their model. Time-domain features of the past CGM data such as minimum, maximum, mean, standard deviation etc. are passed as inputs to the model along with the data, in the hopes of improving accuracy. The approach was tested with the data of 12 different patients, across different PHs of 15, 30, 45 and 60 minutes. The accuracy found varied from patient to patient, with the best patient having RMSE values of 1.42, 3.61, 6.22 and 8.03 mg/dL respectively across the PHs and the worst obtaining values of 4.72, 11.32, 16.88 and 26.65 mg/dL respectively. The accuracy on the best patient is incredibly promising. A clinically acceptable BG prediction is stated to be one within 20% of the reference value [22], which would mean that with a RMSE of 8.03 mg/dL at 60 min PH, and with typical BG levels being between 120 mg/dL and 162 mg/dL [5] this model is providing clinically acceptable and accurate predictions 1 hour into the future. With this in mind, we denote the term of a clinically acceptable prediction error to be one that is within a 20% range of 120, or one that is below 24 mg/dL (1.5 mmol/L). Whilst this model is a great improvement on the basic use of a NN, the difference in accuracy between the best and worst patients highlights the large performance variability - a variability that could result in medical emergencies if used currently without proper in vitro testing.

2.5 RNN Approaches

2.5.1 RNNs

Another method to try and better capture the relationship between previous CGM readings and future expected readings is a specific type of NN called a Recurrent Neural Network (RNN) [23]. An RNN differs from an ordinary 'feedforward' ANN in the sense that in an ANN, all inputs are passed from the input layer 'forwards' through each layer until they reach the output layer. By contrast, an RNN implements a 'loop-like' structure. The inputs are recursively passed into the network such that the value at a node in the network depends on both the current input and a hidden representation, which is calculated by all the previous inputs to the model. In this way, RNNs offer a way to calculate compound representations that depend on all the previous inputs to the network, rather than just passing in and handling the previous CGM data as separate datapoints.
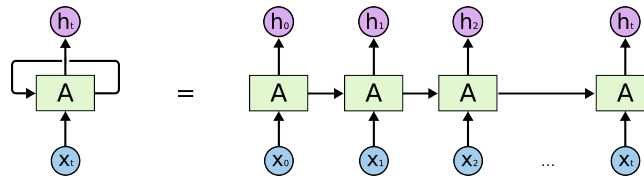
Fig. 5. A diagram showing the 'looped' structure of an RNN [24]

The most often seen implementation of RNNs is the Long Short Term Memory (LSTM) RNN, which, at each node in the network, implements an input, output and forget gate to control the value of the recursive hidden representation [25]. The forget gate controls what information from the previous hidden representation we keep, the input gate controls which information from the current input we add, and the output gate controls what information we pass to the next loop of the and would definitely provide some benefit to T1DM patients network as the next hidden representation. The values of these gates are iteratively updated during the training process to obtain the best accuracy.
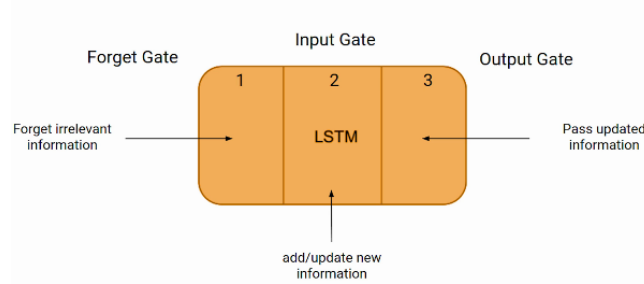

Fig. 6. A diagram showing the functions of the different LSTM gates [25]

These gates control the value of the hidden representation in such a way that it avoids the problems of exploding/vanishing gradient; a major issue with traditional RNN networks, where exceptionally large/small values of gradients when updating weights in the network. This leads to issues when training where either the updates to weights become far too large or far too small to result in any real performance improvement for the network.

2.5.2 RNNs for Blood Glucose Prediction

Robertson et al. [26] used RNNs to perform BG prediction on data generated from the Artificial Intelligence Diabetes Assistant (AIDA) diabetes simulator [27]. AIDA is a freeware diabetes simulator that models steady-state glucose-insulin interactions by describing the physiology of a person with T1DM. This model considers carbohydrate intake, insulin, kidney function and insulin sensitivity. Predictions were made on the data for both short (15-60 minutes) and long term (8-10 hours) PHs. Some points of note viewed from their experiment were that in short term prediction, there were errors between the predicted and actual Blood Glucose Levels (BGL) of over 1 mmol/L - an amount that is not only inaccurate by our measure of accuracy, but also very unsafe in practice. It was also

noted that predictions directly after input events (e.g. meal eaten, insulin taken) were much less accurate; showing that the model wasn't able to correctly handle these events as it should. The model was also seen to be far more accurate at night, most likely due to the lack of events such as eating meals, which would lead to a dramatic change in BGL that the model cannot accurately predict. Overall, in the short term a RMSE of 0.15 ± 0.04 SD mmol/L was viewed and in the long term a RMSE of 0.14 ± 0.16 SD mmol/L was seen - again we note the trends that the model is far more unpredictable at longer term PH. Whilst these results are very promising, and accurate predictions are generated even at a longer term PH, it is also important to consider that the data was obtained in silico from a mathematical model, and so is not hugely indicative of performance on real life data. The mathematical model only handles a few variables to enact changes on its output BGLs, whereas in free-living conditions there are many more factors, such as stress levels and illness, whose fluctuations can impact BGLs in real-life patients.

The model developed by Robertson et al., was limited, performing poorly when significant events are taking place, such as the ingestion of carbs and/or insulin being delivered, which can result in rapid changes in BGL. This is a common issue in the research of BGL prediction and whilst some believe that by studying just trends in the CGM data of a patient to implicitly identify when events like this take place, and change our predictions accordingly, there is a separate approach of using Compartmental Models (CMs) [28] to directly input the data of these events (to include carbohydrate and insulin data) into the network. CMs are mathematical functions that try and describe the processes that occur in the inaccessible parts of the human body. For instance, you would have one model describing how the carbohydrate intake at a given time would impact blood glucose levels over a given period of time afterwards. These CMs can range from simple bell curves to much more complex models such as Hovorka's model [29], and there has been research done into the benefits and drawbacks of using simple or more complex CMs.

Mougiakakou et al. [30] proposed a solution for Blood Glucose Forecasting involving the combination of CMs and RNNs. The input data is passed to three separate CMs, which estimate 1) the effect of short acting insulin intake, 2) the effect of fast acting insulin intake, and 3) the effect of carbohydrate intake on blood glucose levels. The outputs of these three CMs are then passed to the RNN along with past CGM data in order to predict subsequent blood glucose levels. Another point of note in this approach is that the RNN is trained with a Real-Time Recurrent Learning (RTRL) algorithm. Ordinarily, when training a network, you optimise the weights with a learning algorithm on a set of training data, then use those pre-determined optimised weights whenever making a prediction. Conversely, with Real Time Learning (RTL), the weights of the network continue to be improved during use, and each pair of predicted and real BGL values is used to iteratively update the network in real-time. This is thought be very applicable to BGL prediction, as you would be able to initially train a model on a batch of patient data, then continually improve the model as it was being used by the patient; permitting the model to become as personalised and accurate as possible for the user. The data for this model consisted of only 4 or 5 glucose measurements per day, over a period of 70 days, making the data far sparser than that which has been seen in previous

approaches. Overall, the approach obtained a RMSE of 41 mg/dL for predictions for each measurement; an impressively accurate result given the sparsity of the data, but a result that is neither particularly accurate nor clinically acceptable for use in vitro. Mougiakakou et al. [30] concluded that including other information in the model such as sex, age, and number of years a patient has had the condition could improve the performance of the model. Of specific interest was the potential inclusion of more CMs to capture patient physical activity.

The approach seen by Mougiakakou et al. [30] used relatively complex, pre-defined mathematical models to describe the effects of the attributes that the CMs used were modelling. Munoz-Organero et al. [31] attempted a different, more complex approach to the formation of their CMs, by training Deep Neural Networks to predict the impacts of Fast Insulin, Slow Insulin and Carbohydrate Intake on blood glucose levels. The output of these three models would be concatenated and fed as an input along with CGM data to give an overall prediction on the BG variation over the next time step. The deep physiological models were trained using 9 hour windows of data, and attempted to learn the BG variations over the time period, based on the three separate factors highlighted. It was noted that one of the key detriments to this kind of approach is the inconsistency of CGM datasets - some will note down every time a patient has a meal or takes insulin, whereas others will miss datapoints, which can lead to inaccuracies in the model. The models were trained and tested using data from both the AIDA diabetes simulator and the open source D1NAMO dataset [32] (an open-source dataset collected from nine real life T1DM patients). Across a 30 minute PH, the model on the D1NAMO dataset achieved a RMSE of 6.42 mg/dL and with the simulated AIDA data, a RMSE of 3.45 mg/dL. These results show that the model was incredibly accurate over the data, with the AIDA data being the more accurate - as expected due to the added variability of real patient data. However, this accuracy is on data generated by healthy patients, and the dynamics of insulin/carbohydrate absorption would change massively due to factors such as stress and illness. This means that, in practice, such complex deep physiological models could become very inaccurate during these times of imperfect health, and thus still are unproven and unsafe for use by real T1DM patients. It would require more real-life testing to verify whether this approach could feasibly work. Unfortunately, in practice, it is very hard to perform real life testing due to a number of factors such as obtaining ethical approval, finding a volunteer pool large enough to give accurate results and making sure that those volunteers adhere to the rules of the study correctly.

Whilst directly modelling the impacts of other factors such as carbohydrate intake with CMs is possible, there is another train of thought that we should be able to intrinsically detect these events and their impacts by looking at different signals in the CGM data. One example of this approach was taken by Wang et al. [33] who combined Variational Mode Decomposition (VMD) with an LSTM RNN optimised by Improved particle Swarm Optimisation (IPSO) to perform BG prediction. Imagine, for example, that a given T1DM patient eats lunch between 12-2 pm every day - we would expect to see a spike in their BG levels around this time of day every day. Through signal decomposition techniques such as VMD, we can extract the separate signals that combine to make the overall BG graph - in this case, one such signal would be a signal with a period of 1 day and a spike

around the lunch time of the patient. Through this decomposition, it is thought to be possible to reduce the non-stationarity of the CGM data, and make it possible for the RNN to make more accurate predictions. To ensure the LSTM has optimal hyper parameters, Wang et al. [33] employed an Improved Particle Swarm Optimisation algorithm - an algorithm that uses many 'particles' initialised at random points in the parameter space, then iteratively converges to the optimal solution by tracking the best particles in the current space, and initialising the next round of particles to be closer to these best ones.

To measure the success of their approach, Wang et al. [33] used both the RMSE between predicted and actual BG levels, as well as the Clarke Error Grid (CEG) [22]. The CEG is a tool for checking the accuracy of blood glucose monitors. It splits predictions into 5 separate zones: A, B, C, D and E. Zone A is defined as clinically accurate; Zone B indicates an incorrect, but benign, and therefore clinically acceptable prediction. Zone C represents values leading to inappropriate treatment, but without dangerous consequences for the patient. Zone D represents values leading to potentially dangerous failure to detect hypo or hyper-glycemic events and Zone E represents values leading to treat hypoglycemia instead of hyperglycemia and vice-versa. For a model to be clinically accurate, predictions should occur within zones A and B. Across a 30, 45 and 60 minute PH the model achieved an RMSE of 3.031, 5.432 and 5.716 mg/dL respectively. At the 60 minute PH, the model had 95.4% of predictions in Zone A, 3.8% in Zone B and 0.8% in Zone D of the CEG. These results at are very promising, but the model is so far untested at the more important very long term PHs that are currently struggling to be accurately predicted over.
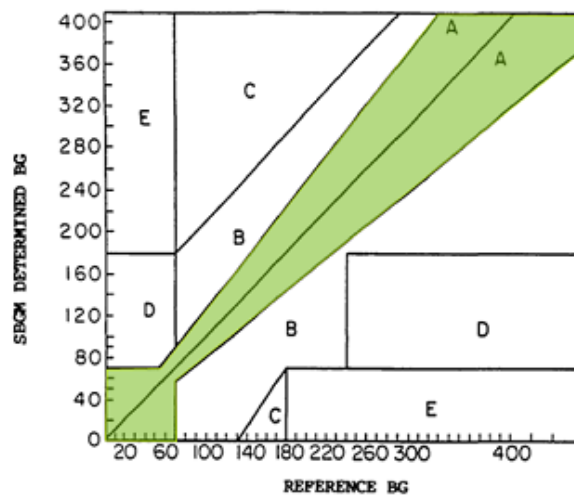

Fig. 7. A visualisation of the CEG [34]

The approaches we have seen so far have all suffered from one common issue - a lack of data to train models on. In typical machine learning problems, datasets often consist of millions of datapoints which allows the creation of very complex and accurate models. In contrast, CGM datasets are difficult to create, and often consist of only thousands of datapoints which leads to a distinct shallowness in the data and inhibits the training of more accurate models. Aliberti et al. [35] sought to rectify this issue and instead of just training their model on the data of one patient, they trained

their model on the data of many patients - whilst keeping a large chunk of the training data as the data of the main patient in question. They claimed that this would not only make more data available for the training of the model but also make their model more flexible. Past approaches that are only trained on a single patient's data require a lot of fine tuning and produce a model that is rigid and can only be used on one specific T1DM patient. In medical practice, it would be infeasible to produce an individual model for every T1DM patient and, as such, Aliberti et al. [35] claimed that their model, which would be trained across many different datasets, could offer a solution to this issue. They tried their attempt using both a Non-Linear Autoregressive NN (NAR), a network which computes the value of a signal y at time t using n past values of y as regressors, thus allowing it to model the dependency between sequential datapoints, and a LSTM RNN. Over 30, 45, 60 and 90 minute PHs, the NAR achieved RMSE of 18.2, 25.31, 33.12, 47.64 mg/dL respectively, and the LSTM received an RMSE of 5.93, 7.18, 13.21 and 28.57 mg/dL respectively. The LSTM was far more accurate than the NAR, and over a short PH this approach performed very well, but once again the performance very quickly dropped off at longer PH.

2.6 Alternative Approaches

So far all of the approaches have viewed the task of BG Prediction as a regression task where you are trying to predict a value for the BG variation over the PH. Zhu et al. [36] instead formulated the problem as a classification problem, where the change between current glucose value and the future glucose value is quantised into 256 different classes to be predicted between. The data was sourced from the OhioT1DM dataset [37], which is a dataset of real-life diabetes patients. Zhu et al. used a Convolution Neural Network (CNN) [38] a type of neural network typically used in image recognition, due to its ability to reduce the dimensionality of input data. Over a 30 minute PH, the approach achieved an average RMSE of 21.72 mg/dL for its predictions, which is not as promising as other approaches we have seen involving regression via an RNN. Another approach not using the traditional RNN was experimented with by Wang et al. [39], who built a model based on Light-GBM [40] - a gradient boosting algorithm that generates a decision tree to make predictions on the data. Over a 30 minute PH, they achieved a RMSE of 0.7721 mmol/L, which just falls within the range of accurate predictions, showing that the approach has merit at shorter prediction horizons.

2.7 Conclusions

In summary, we have seen a variety of different approaches to performing BG prediction using neural networks with various levels of success. Some approaches incorporated a large number of factors as well as CGM data, whilst others included more factors relating to the CGM data itself such as maximum and minimum points. However, despite all of these differences, all of the approaches shared in common that the performance dropped of quickly as the PH grew longer as well as the

fact that they had not been trialed at all in an in vitro scenario. Coupling this lack of in vitro testing, along with the fact that approaches used a myriad of different datasets makes it very hard to say whether one approach is better than another, despite the common RMSE metric, or whether the dataset they were using lent itself towards being much easier to be predicted over; either due to a more gradual nature in the change of BG levels, or a better quality of data.

To rectify the second of these issues, some of these researched approaches and their features will be selected and implemneted using a consistent dataset - the OhioT1DM dataset [37]. Through the use of a common dataset, there will be a common platform by which we can judge the performance of these different methods over differing prediction horizons.

3 Methodology

3.1 Introduction

The flow chart below details the main aspects of the project and the pipeline in which they fit together. The data is first read from the OhioT1DM dataset using pandas before multiple levels of preprocessing occur to generate new signals from the input data. This new data is then split into windows based on the PH being predicted over, and models are built then optimised on the data. Finally, the models are evaluated using a number of metrics and visualisations are produced from these different metrics.
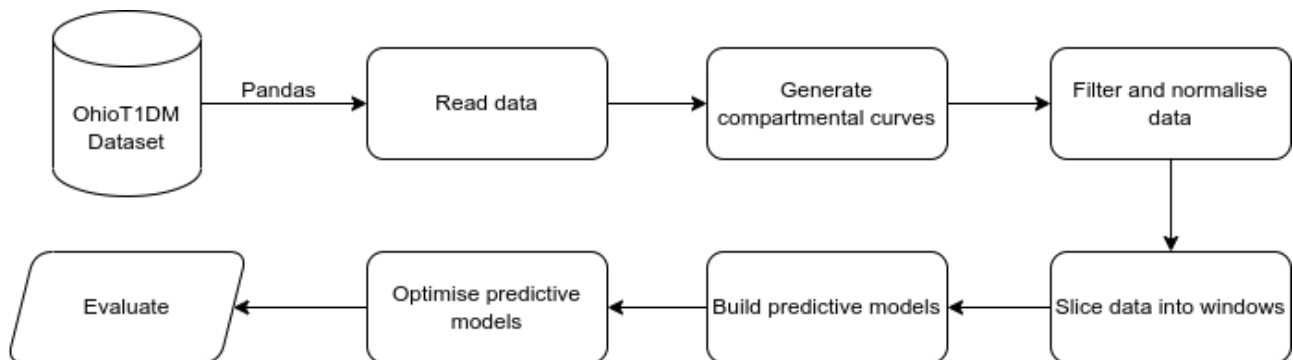

Fig. 8. A flow chart detailing the main parts of the project

To implement the project, a python3 jupyter notebook development environment was used. A variety of different libraries were used, the most prevalent of which being listed below:

- pandas

- numpy

- tensorflow

- matplotlib

- sklearn

- pickle

- keras tuner

3.2 Detail

Tell story of the development of the models, from data processing to choices made in model hyper-parameters / cleaning etc.

3.3 OhioT1DM Dataset

The OhioT1DM [37] dataset is a dataset developed to assist in the research of blood glucose level prediction. It consists of 8 weeks of CGM data taken every 5 minutes for 12 different patients with T1DM. Alongside the CGM data are a number of events of interest such as when a patient had a meal and the amount of carbohydrate consumed, or when a patient injected themselves with a dose of insulin. There are 20 such factors included in the dataset. The data for the patients is split into training examples and testing examples and for each patient there are 13,000 samples split across the two sets with / 10,500 and 2,500 samples being in the training and testing sets respectively. This accounts for a 24% train-test split in the data. The patient data is held in XML files and approval to use the dataset was obtained through The University of Manchester.

The data in the dataset was gathered from two separate cohorts - one cohort in 2020 and the other in 2018. For the purpose of this project, we will be using the data from 3 randomly selected patients from the 2018 cohort. Information about the selected patients can be seen in the table below.

Table 1. A table showing individual information about the selected patients

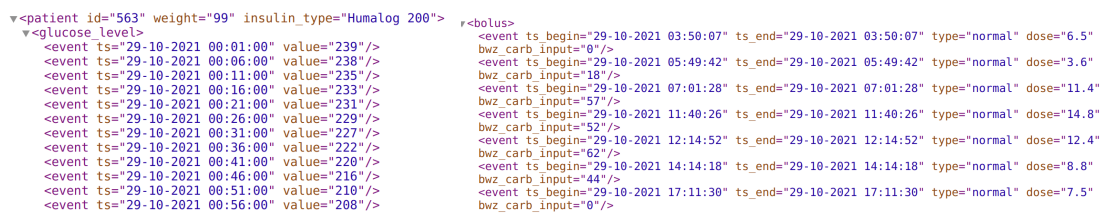| ID	Gender	Age	Pump Model	Sensor Band
563	Male	40-60	530G	Basis
570	Male	40-60	530G	Basis
588	Female	40-60	530G	Basis

As we can see from this table, there are two male patients and one female patient; with all of the patients being between 40-60 years of age. The model of the insulin pump that the patients used

as well as the type of sensor band (a sensor used to record data such as heart rate) are also included for each patient. These data, whilst useful to fine-tuning predictions made on their respective fields were ignored for this experiment as no data relating to the sensor band was utilised, and fitting insulin curves to the specific pump model was deemed to reduce the flexibility of the model by specifying it to one kind of pump.

3.4 Reading in the Data

3.4.1 Data storage format

As mentioned earlier, all of the data for the patients is stored in XML files. Each individual field of data for the patient (e.g. the CGM data) is stored within an individual object that consists of a list of events of a time when the event occurred, and a value for the event. There are separate data files for both training and testing data, so all reading processes had to be performed twice.

```
▼<patient id="563" weight="99" insulin_type="Humalog 200">   ▼<bolus>
  ▼<glucose_level>                                              <event ts_begin="29-10-2021 03:50:07" ts_end="29-10-2021 03:50:07" type="normal" dose="6.5"
    <event ts="29-10-2021 00:01:00" value="239"/>              bwz_carb_input="0"/>
    <event ts="29-10-2021 00:06:00" value="238"/>              <event ts_begin="29-10-2021 05:49:42" ts_end="29-10-2021 05:49:42" type="normal" dose="3.6"
    <event ts="29-10-2021 00:11:00" value="235"/>              bwz_carb_input="18"/>
    <event ts="29-10-2021 00:16:00" value="233"/>              <event ts_begin="29-10-2021 07:01:28" ts_end="29-10-2021 07:01:28" type="normal" dose="11.4"
    <event ts="29-10-2021 00:21:00" value="231"/>              bwz_carb_input="57"/>
    <event ts="29-10-2021 00:26:00" value="229"/>              <event ts_begin="29-10-2021 11:40:26" ts_end="29-10-2021 11:40:26" type="normal" dose="14.8"
    <event ts="29-10-2021 00:31:00" value="227"/>              bwz_carb_input="52"/>
    <event ts="29-10-2021 00:36:00" value="222"/>              <event ts_begin="29-10-2021 12:14:52" ts_end="29-10-2021 12:14:52" type="normal" dose="12.4"
    <event ts="29-10-2021 00:41:00" value="220"/>              bwz_carb_input="62"/>
    <event ts="29-10-2021 00:46:00" value="216"/>              <event ts_begin="29-10-2021 14:14:18" ts_end="29-10-2021 14:14:18" type="normal" dose="8.8"
    <event ts="29-10-2021 00:51:00" value="210"/>              bwz_carb_input="44"/>
    <event ts="29-10-2021 00:56:00" value="208"/>              <event ts_begin="29-10-2021 17:11:30" ts_end="29-10-2021 17:11:30" type="normal" dose="7.5"
                                                               bwz_carb_input="0"/>
```

Fig. 9. An image showing a short extract of CGM data and insulin injection data for patient 563

3.4.2 Time Formatting

To read in the data, the `pandas.read_xml()` function from the pandas library is used. As the 'ts' field of the event is currently just a string, the field was also reformatted to become a pandas datetime object. As all of the CGM data is taken at five minute intervals, but the data of other events (such as insulin being injected) is recorded at random time frames, we round all time events to the nearest five minute point, allowing each individual event to be matched exactly to a CGM data point being recorded. For any field where two events ended up being rounded to the same 5 minute point, their values were summmed to represent the combination of both events. For all events that had a 'ts_begin' and 'ts_end' property, the 'ts_begin' property was taken to represent the time of the event occurring. Once read in, all of this data was stored into a pandas dataframe.

3.5 Generating Compartmental Curves

The data now stored in the dataframe is a very sparse dataset. As there are around 13,000 samples of CGM readings, and only 30 samples of meal events for example, the majority of the columns in

the dataframe are filled with None as the value. To make these columns useful for prediction by a model, it is necessary to make the discrete events into a continuous data curve that spans the entirety of the period that there are CGM readings for.

3.5.1 Differential Equations

It is pertinent to note that from the OhioT1DM dataset two of the insulin events looked at use two different types of insulin. The first is short-acting insulin, which is typically taken around 25 minutes before a meal, takes about 30 to 60 minutes to start working and lasts from 5 to 8 hours [41]. The second type of insulin is basal insulin, which is a type of background insulin that is usually adjusted once or twice a day. It is released slowly throughout the day to deal with any sugars that the body may be making or moving. Both types of insulin are used as additional data points when making later predictions.

To model the effects of the different events over time, we refer to the Haiya et al. model [42], a mathematical model for describing the rate of absorption of glucose and insulin into the body. The dynamics of insulin and glucose are modelled by a delay differential equation [43] model, which is then broken down into two separate piecewise linear functions.

$$
G_{\mathrm{in}}(t) = \begin{cases} 0.05 + \frac{5}{15}t, & 0 \leqslant t < 15 \ (\mathrm{min}), \\ 0.05 + 5\frac{45-t}{45-15}, & 15 \leqslant t < 45 \ (\mathrm{min}), \\ 0.05, & 45 \leqslant t \leqslant 240 \ (\mathrm{min}). \end{cases}
$$

$$
I_{\mathrm{in_{regular}}}(t) = \begin{cases} 0.25, & 0 \leqslant t \leqslant 30 \ (\mathrm{min}), \\ 0.25 + 1 \cdot (1 + \frac{t-120}{90}), & 30 \leqslant t < 120 \ (\mathrm{min}), \\ 0.25 + 1 \cdot (1 - 0.5\frac{t-120}{120}), & 120 \leqslant t < 240 \ (\mathrm{min}), \\ 0.25 + 0.5 \cdot (1 - \frac{t-240}{240}), & 240 \leqslant t \leqslant 480 \ (\mathrm{min}). \end{cases}
$$

Fig. 10. Piecewise linear functions for both glucose dynamics and short-acting insulin dynamics [42]

In these equations, G_{in} and $I_{in_{regular}}$ are the volume of glucose and insulin absorbed respectively with respect to the size of the dose, and t is the time in minutes since the event took place. It is then possible to make the discrete events of insulin injection and carbohydrate intake continuous by applying these piecewise linear functions to every event occurrence in the dataframe. All values within the dataframe which still have a value of None are set to 0 to indicate that no insulin or glucose is being absorbed at that instance in time.

3.5.2 Insulin Curves

As the data samples within the dataframe are split at 5 minute intervals, and each insulin event results in insulin being absorbed for the next 480 minutes, we take each event to impact itself and the next 95 datapoints within the dataframe. Using the amount of insulin as the initial dose to control the area under the curve, each indiviudal event produces a curve like the one seen in the following figure.
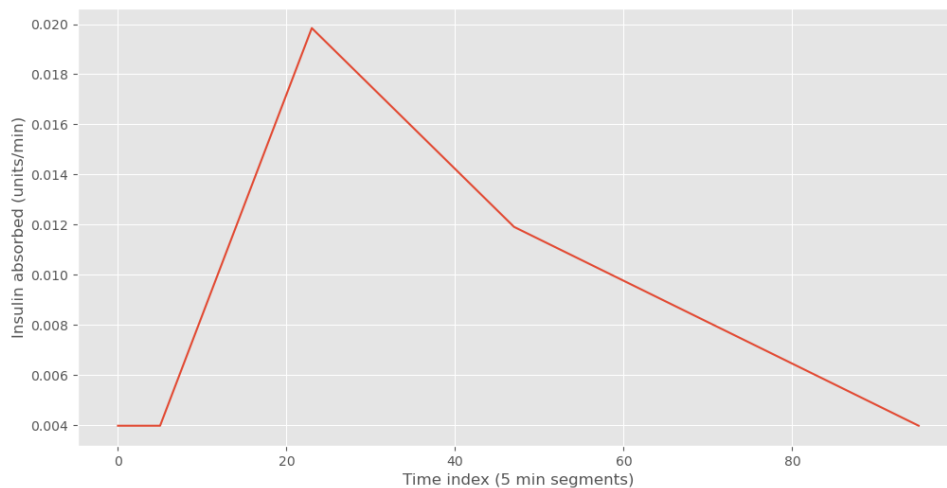

Fig. 11. A graph showing the absorption curve for a dose of insulin

To account for overlapping periods of insulin absorption, we first calculate the amount of insulin being absorbed at each point in the curve, then sum this back onto the original dataframe. This ensures that any insulin doses which impact the same points in time of the data both contribute to the volume of insulin being absorbed. This is then done for every insulin dosing event in the dataset, and produces a singal for insulin absorption. The insulin absorption signal for the testing data of patient 563 can be seen in the figure below.
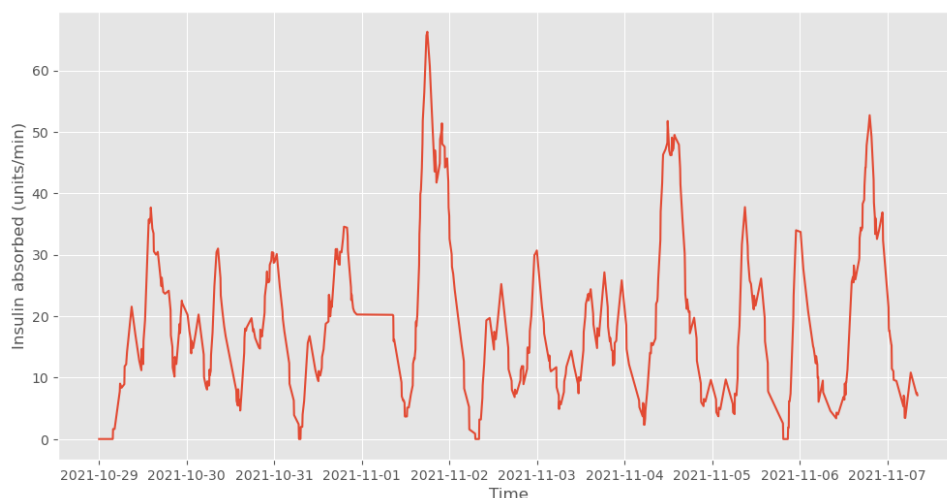

Fig. 12. A graph showing the insulin absorption curve for the testing data of patient 563

By expanding the size of the continuous signal and plotting it alongside the discrete signal, the relationship between the two can be seen. It is of note that when the discrete signal has a large value, the continuous signal begins to climb before dropping off.
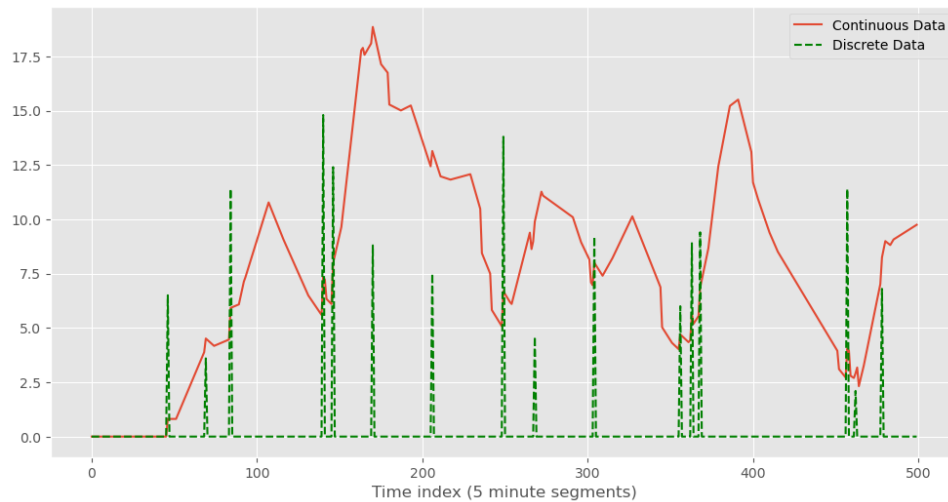

Fig. 13. A graph showing the overlap of the discrete and continuous insulin absorption signals for a slice of the testing data of patient 563

This signal can now be used alongside the CGM data as a CM when making models that predict using multiple points of interest from the patient dataset.

3.5.3 Carbohydrate Curves

The same process as above is performed for carbohydrate intake, but only 48 datapoints are involved due to the shorter length of time to absorb glucose.
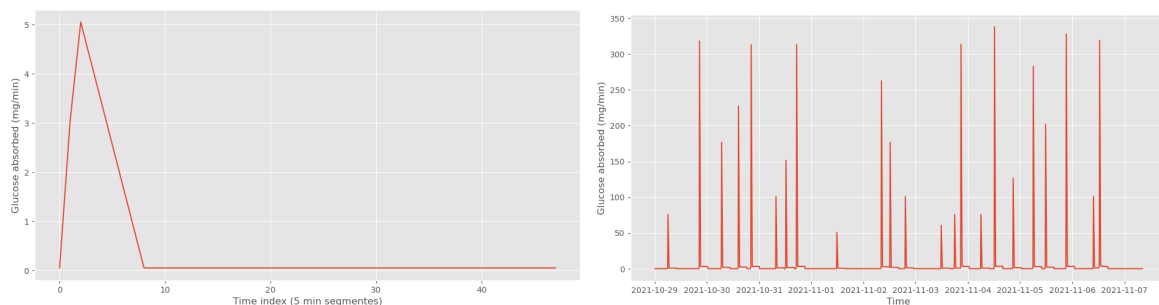

Fig. 14. Graphs showing an individual carbohydrate absorption curve and the function applied to the meal data of patient 563

This curve can now also be used for predictions.

3.5.4   Basal Graph

Whilst the events for carbohydrate intake and insulin injection signified a point of drastic change, the events for the basal field simple state that the rate of continuous infusion of the background basal has been adjusted. Due to this, to make the signal continuous, simply 'fill in the gaps' by setting the value of the field to the current basal rate until the basal rate is changed, at which point the value used to 'fill in' is updated to this new value. A patient will often have a pattern where they consistently set their basal rate to values at certain points in the day, which leads to the periodicity seen in the following figure.
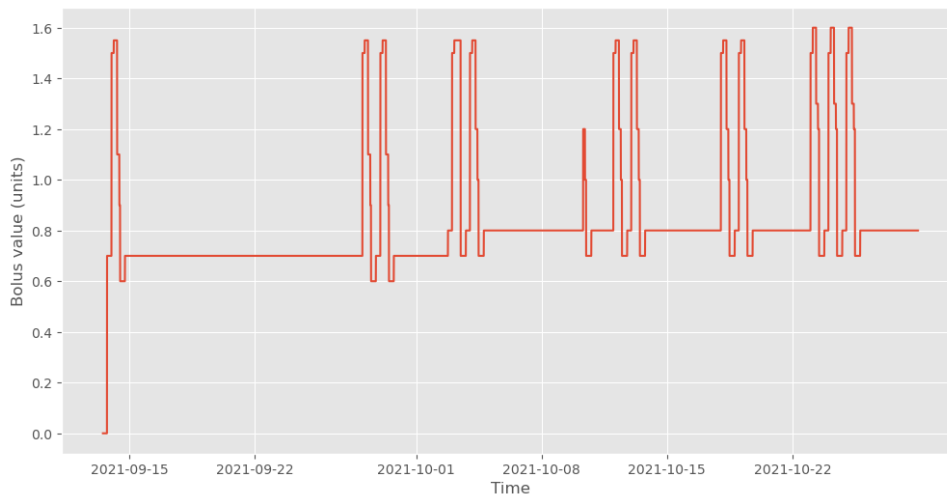

Fig. 15. Graphs showing the curve generated from the basal events of patient 563

The large gap that can be seen between the first and the second peak is most likely due to the patient recording the data forgetting to record the changes to their basal rate in the dataset.

The dataframes with these newly generated curves were then saved to csv format using pandas in order to be read into separate notebooks when making predictions on the data.

3.6 Filtering and Normalisation

3.6.1   Filtering

As the measurements for CGM data are based off of interstitial fluid [ref about measurements], there is much noise present in the generated data which can negatively affect the ability of ML models to accurately fit the data. As such, it is common to apply filtering techniques to the data in an attempt to reduce this noise and improve model performance.

In this project, Savitzky-Golay smoothing [44] was used, which is a filtering technique that increases the precision of a given singal, without reducing the signal tendency. This is done through a process

known as convolution [45], by fitting successive windows of data points with a polynomial. Both the size of the window and the degree of the polynomial can be controlled as inputs to the filter. It is common to use a low-degree polynomial when performing the filtering. The following figure demonstrates the effects of the Savitzky-Golay filter with a window size of 7 data points and a cubic polynomial.
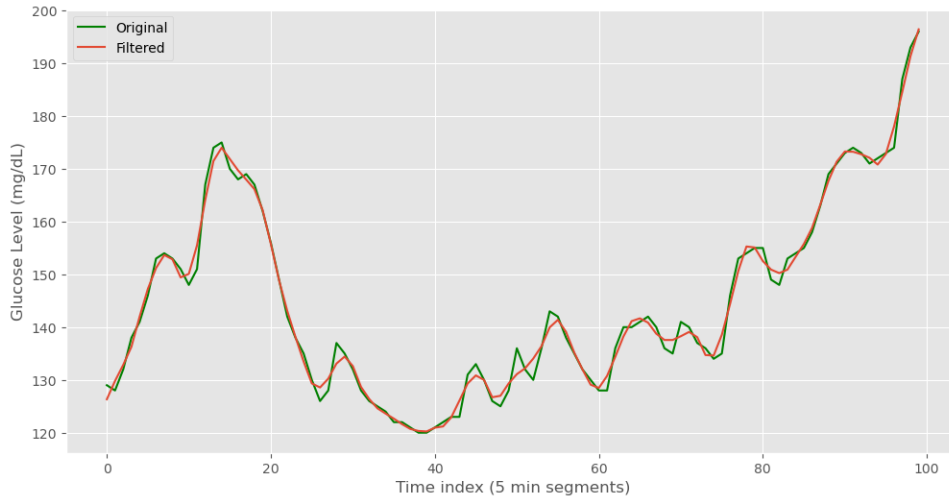

Fig. 16. A graph showing the effects of Savitzky-Golay smoothing

When experimenting with the trained models, the performance is compared on both filtered and unfiltered data, so the impacts of this filtering technique can be viewed. It is important to note that only the training data is filtered, and all predictions are made on the original testing data, to ensure we don't skew the results.

3.6.2   Normalisation

To ensure that all of the time series data used is on the same scale, it is important to normalise the data before training any models on it. This normalisation process improves both the training process and the final performance of the model, as larger numbers will have a more noticeable impact on the model's training and may lead to bias in the model placing too much importance on one input rather than the others. In the case of this project, the CGM data has inputs that can go up to 400, whilst the input for basal data never goes above 2. This makes normalisation even more crucial to avoid bias between these two inputs.

In this project, Min-Max scaling was used [46], which is a normalisation technique that rescales all input features to values between the range [0,1]. The process is completed by applying the following function to each of the input features:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

where x is the input timeseries data, and x_{min} and x_{max} are the min and max of the data respectively.

In this way, all inputs are scaled to the desired range, whilst still keeping the relative differences between the values of the original input.

	GlucoseLevel	Long	Short	Carb		GlucoseLevel	Long	Short	Carb
count	12124.00000	12124.000000	12124.000000	12124.000000	count	12124.000000	12124.000000	12124.000000	12124.000000
mean	146.07712	0.830823	12.535700	6.399876	mean	0.294659	0.519264	0.168706	0.016897
std	49.68904	0.250614	9.056443	26.727100	std	0.138025	0.156633	0.121882	0.070567
min	40.00000	0.000000	0.000000	0.000000	min	0.000000	0.000000	0.000000	0.000000
25%	108.00000	0.700000	6.450000	0.000000	25%	0.188889	0.437500	0.086804	0.000000
50%	140.00000	0.800000	11.035417	0.000000	50%	0.277778	0.500000	0.148515	0.000000
75%	177.00000	0.800000	17.000434	1.700000	75%	0.380556	0.500000	0.228792	0.004488
max	400.00000	1.600000	74.305208	378.750000	max	1.000000	1.000000	1.000000	1.000000

Fig. 17. Tables showing the training data for patient 563 before and after min-max scaling

In the above tables, Long, Short and Carb refer to basal data, insulin dose data and meal data respectively.

From these tables, the impacts of min-max scaling can clearly be seen. Looking at the GlucoseLevel column, where before the min and max were 40 and 400, they are now 0 and 1 respectively. However, by looking at the mean, we can see the relative differences between the values of the datapoints has been preserved. The new mean value of 0.294659, can be interpreted as saying that the mean is 29% of the way between the minimum and max value. Applying this to the original data, by calculating $(400 - 40) * 0.294659 + 40$ to see the value 29% of the way between the min and max, the value 146.077 is obtained, which is the same as the mean in the original table.

Once again, the models are trained on both normalised and un-normalised data so that the effects of this technique are clearly visible.

3.7 Range Slicing

With the data processing complete, it is now possible to make predictions on the data. The notion of a prediction range is defined to be a certain configuration of the dataset that we wish to predict over. Each prediction range is defined by both the number of past datapoints that we use to inform predictions, as well as the PH that is being predicted over. The prediction range is created using tensorflow's `keras.utils.timeseries_dataset_from_array()`, which enumerates over the timeseries data and splits the data into lots of separate samples of a set of input datapoints and a datapoint to be predicted. This process can be seen in the following figure where the blue sections represent the input data into the model, and the green sections represent the corresponding data point to be predicted.
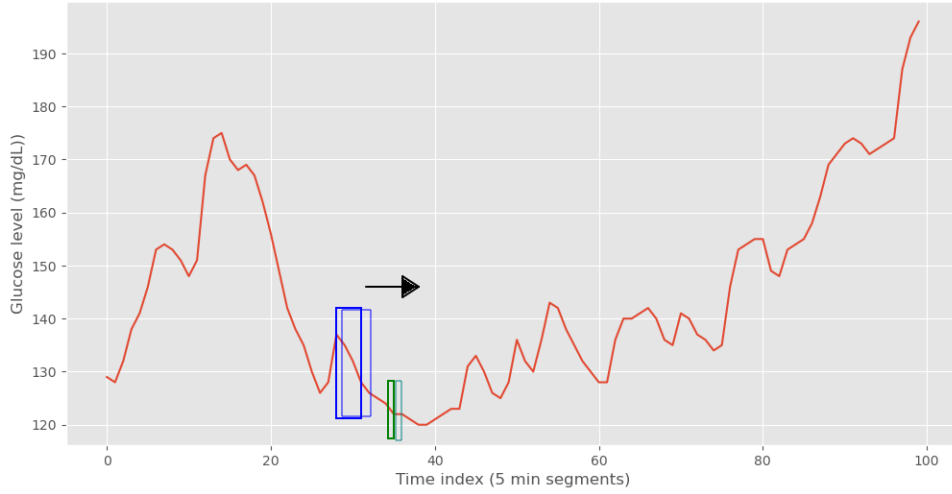
Fig. 18. A visualisation of the prediction range generation process

The following figure is produced by zooming into one of the blue input sections and its corresponding green prediction point. It features three separate slices for a prediction range using 20 minutes of past data, predicting over a PH of 15 minutes into the future.
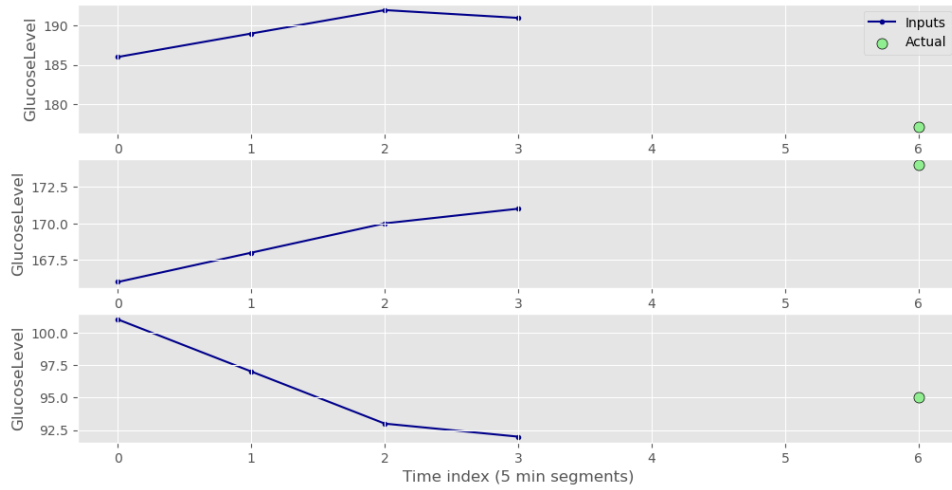

Fig. 19. A visualisation of individual slices of a prediction range

Such prediction ranges were generated using 20 minutes of past data over PHs of 15,30 and 60 minutes for each of the selected patients.

3.8 Building the Models

All models for this project were built using tensorflow [47], which is a python library for building industry grade machine learning models.

3.8.1 Baseline

When performing machine learning experiments, it is common to have a baseline to test model performance against. In the case of this project, we define our baseline to be a linear baseline. That is to say that the baseline is a model which takes the last seen value from the input data and uses that value as the value for the prediction. A visualisation of this baseline can be seen below.
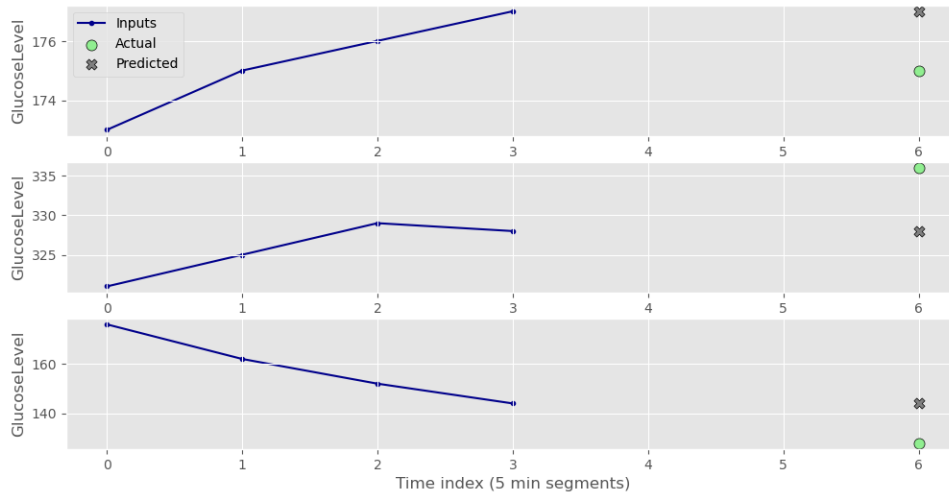

Fig. 20. A graph showing how the linear baseline makes predictions over a 15 min PH

3.8.2 Basic LSTM

The set of approaches being explored in this project all use the basis on an LSTM RNN model, which was explained in chapter 2.5.1 of the literature review. Using `tensorflow.keras.layers.Sequential`, which allows the creation of feedforward NNs, a neural network is defined that consisconsiststs of one LSTM layer created with `tensorflow.keras.layers.LSTM` followed by one Dense layer made with `tensorflow.keras.layers.Dense`. The meaning of the dense layer is that every node in the previous layer is connected to every node in the current layer. In this network, the Dense layer consists of only one node, and serves as the output layer for the model which returns one prediction for the forecasted BG value. Finally, before being returned by the model, the output is reshaped to obtain the correct dimensions to be used by the rest of the program.
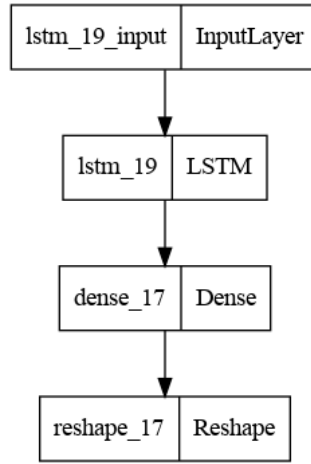
Fig. 21. A visualisation of the structure of the basic LSTM

The number of nodes in the LSTM layer is left as an optimisation task.

All of the LSTM networks were trained by minimising Mean Squared Error (MSE) through the iterative process of gradient descent - which was described in the literature review. MSE is very similar to RMSE in that it calculates the mean of the square of the differences, but unlike with RMSE, we don't take the square root.

$$MSE = \frac{\Sigma_{i=1}^{N}\Big(Predicted_i - Actual_i\Big)^2}{N}$$

To optimise the gradient descent process, Adaptive Moment Estimation (Adam) [48] optimiser was used. Gradient descent with Adam optimisation differs from normal gradient descent in the sense that with normal gradient descent, there is a constant learning rate that defines weight updates, whereas in Adam optimised gradient descent, each weight in the network has its own individual learning rate which is changed during the training process. This adaptive changing of the learning rate speeds up the rate at which the gradient descent algorithm converges on an optimal weight configuration.

3.8.3 Residual Models

Whilst the previous model attempted to predict a straight value for the BG level, there is another strategy often used in timeseries prediction which is to make use of residual connections. By understanding that there is unlikely to be a large amount of change between the last input point and the prediction point, we instead try to predict the change between the two points, rather than just trying to predict the prediction point. This can result in a faster training process as well as better performance.

In practice, this is implemented using tensorflow. We wrap the LSTM in a custom 'change layer' which takes the output of the LSTM, adds it to the last value of the input CGM data and then returns the summed value as the new output of the network. Training is performed in the same manner as the basic LSTM model.

3.8.4   Bidirectional LSTM

Using the base LSTM structure, it is also possible to construct more complex models. One example of this is the Bidirectional LSTM model, which combines two traditional LSTM networks. Into one of the LSTM layers, the input is fed normally into the model - this is called the 'forward layer'. Into the other, the input is fed backwards into the network - this is called the 'backwards layer'. The representation generated for each input into the network is the concatenation of its representation in the forwards and backwards layers. in this way, Bidirectional LSTM networks are able to not only model the impact that previous BG levels have on future BG levels in the input data, but also the effect that future BG levels have on the previous once (due to the input reversal).
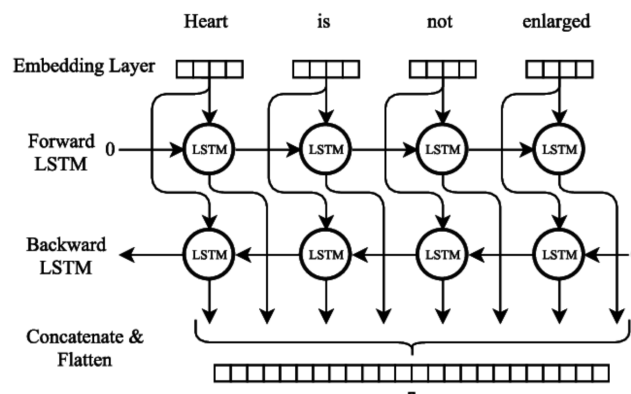
Fig. 22. A visualisation of a Bidirectional LSTM [49]

As with before, this model was built using tensorflow. First a forward and a backward layer are defined as individual `tensorflow.keras.layers.LSTM()` layers, then combined together into a `tensorflow.keras.layers.Bidirectional()` layer. The same Dense layer and Reshape layer as in the earlier basic LSTM are also included in this model.

Fig. 23. A visualisation of the structure of the Bidirectional LSTM

3.8.5 Multi-Input Models

Concerning models that also include meal data, insulin data and basal data, the main difference in the creation process lies in the generation of the prediction range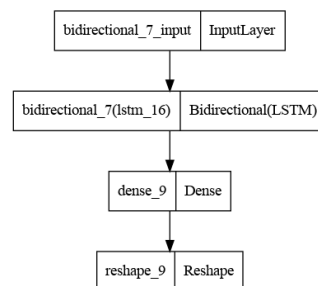. Instead of just slicing the CGM data into chunks that are related to a future point to be predicted, all of the generated curves are also sliced such that the predictio range consists of [CGM data, Insulin Data, Meal Data, Basal Data] -> Predicted point. The structure of this when combined with the model can be seen in the following figure.
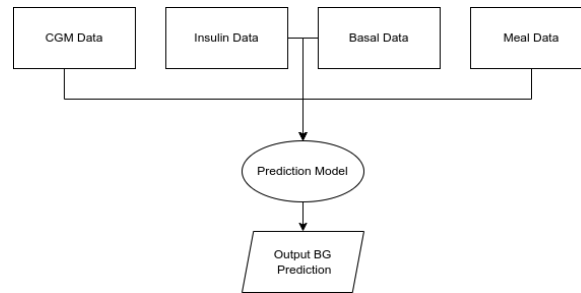

Fig. 24. A visualisation of the use of the compartmental curves

The basic LSTM structure as well as the more complex Bidirectional structure are both used with multiple inputs to see what, if any, improvements are made with the inclusion of more data.

3.8.6 Metrics

To measure the performance of the models, two main metrics will be used - the RMSE of the predictions, and CEG analysis on the predictions.

The implementation of RMSE is simple as tensorflow provides a ready-made implementation in `tensorflow.keras.metrics.RootMeanSquaredError()`. However, when using data that has been normalised with min-max scaling, it is not as trivial. The numbers in the normalised data all range between 0 and 1, so finding the RMSE of predictions in this range is not informative at all as to the actual performance of the model - all error values will be tiny. Instead, the scale used to normalised the data is kept track of when normalising, and the inverse transformation is applied to both the actual and predicted value from the normalised data before calculating the RMSE value. This is done by writing a custom metric in tensorflow, and passing this metric to the model to be trained.

To calculate CEG accuracies, an implementation of the CEG, based on [50], is written, which returns the percentage of prediction points that occur in each of the zones of the grid.

3.9 Model Optimisation

Parameters within neural networks such as the learning rate and the number of nodes in each layer of the network are commonly known as hyperparameters, and should be optimised to obtain the best performance out of a model.

Finding optimal hyperparameters for the models was done using the keras tuner, with the algorithm used being hyperband tuning [51]. Hyperband tuning is a type of tuning algorithm that relies on early-stopping to quickly find hyperparameter values that give good performance. A large number of models are trained for a few epochs, with only the best performing models being carried forward after each round. These rounds continue until only the best-performing set of hyperparameters remains.

3.10 Summary

All of the trained models were saved using tensorflow, so that they could be loaded directly into a python jupyter notebook environment. With this loading capacity, all pre-processing and necessary development was complete. It was now possible to begin experimenting with the different models so that their performances could be compared with one another, and with different approaches in literature.

4 Experimentation and Results

4.1 Introduction

The experimentation for the project was broken up into 3 main sections, which are listed below:

1. Basic LSTM prediction vs Residual LSTM prediction on unfiltered and filtered data.

2. LSTM with compartmental data on normalised and unnormalised data.

3. Bidirectional LSTM and other approaches.

After all of the experimentation is done, a general discussion of the results over all 3 separate sections of the experimentation will also take place.

4.2 Detail

4.3 Baseline Performance

To give a baseline to test model performance against, we first test the linear baseline on all of the prediction ranges generated and record the RMSE. All RMSE values are measured in mg/dL.

Table 2. A table showing the RMSE performance of the linear baseline model

Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
13.05	20.78	33.96	12.75	22.06	36.07	11.44	19.46	32.21

From the better accuracy on patient 588 in this table, we can deduce that the data of patient 588 has much more gradual changes in BG levels, as these gradual changes are the ones that the baseline will predict more accurately on.

4.3.1   Optimisation

When optimising with hyperband tuning, different optimal parameters were found for each of the different prediction horizons. Over a 15 minute horizon, 40 training epochs a 0.001 learning rate and 288 units in the LSTM layer were used. For a 30 minute PH, 36 training epochs, 224 units and 0.001 learning rate were found as optimal. Finally, for the 60 min PH, 23 training epochs, 352 units and 0.001 lr were found to be the optimal settings. These model settings were carried over for all of the patients, but the hyperband tuning was completed on the data of patient 563.

4.3.2   Basic LSTM vs Residual LSTM

The difference between the strategy of making residual predictions versus straight predictions was first experimented with. We train a model with prediction ranges using 20 minutes of past data and a PH of 15 mins, 30 mins and 60 mins. Both data smoothed by Savitzky-Golay filtering and un-smoothed data are used. The RMSE between predicted and actual data is recorded and the results can be seen in the tables below. Each model was trained 3 times, with the average RMSE performance being recorded.

Table 3. A table showing the RMSE performance of the basic LSTM model

Unfiltered Data								
Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
12.60	18.95	30.56	10.74	17.95	29.72	11.11	20.63	33.20
Filtered Data								
Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
14.62	19.98	30.81	10.77	17.58	31.21	12.39	19.56	32.88

Table 4. A table showing the RMSE performance of the residual LSTM model

Unfiltered Data								
Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
14.18	20.67	32.84	11.87	18.61	31.28	13.23	21.09	34.65
Filtered Data								
Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
14.36	21.09	31.58	11.47	18.01	31.49	13.31	20.35	33.92

What is made immediately obvious by these tables, is that the basic LSTM performs noticeably better than the residual LSTM in making predictions. This could be due to the fact that sometimes the change between the last value and value to be predicted is large, which affects the residual models ability to converge on an accurate setting of weights.

The performance of the models on the flitered data is much harder to judge, with the filtered data sometimes performing better and sometimes performing worse. What is interesting to note is that the training error was lower in all cases when using the filtered data. This shows that the filtering technique did work, as the model was able to fit better to the data which shows that the noise was reduced. However, perhaps the smoothing was too strong and would have been better done with a lower window size as clearly the models sometimes also overfit on the training data which explains the fact that sometimes the performance is worse.
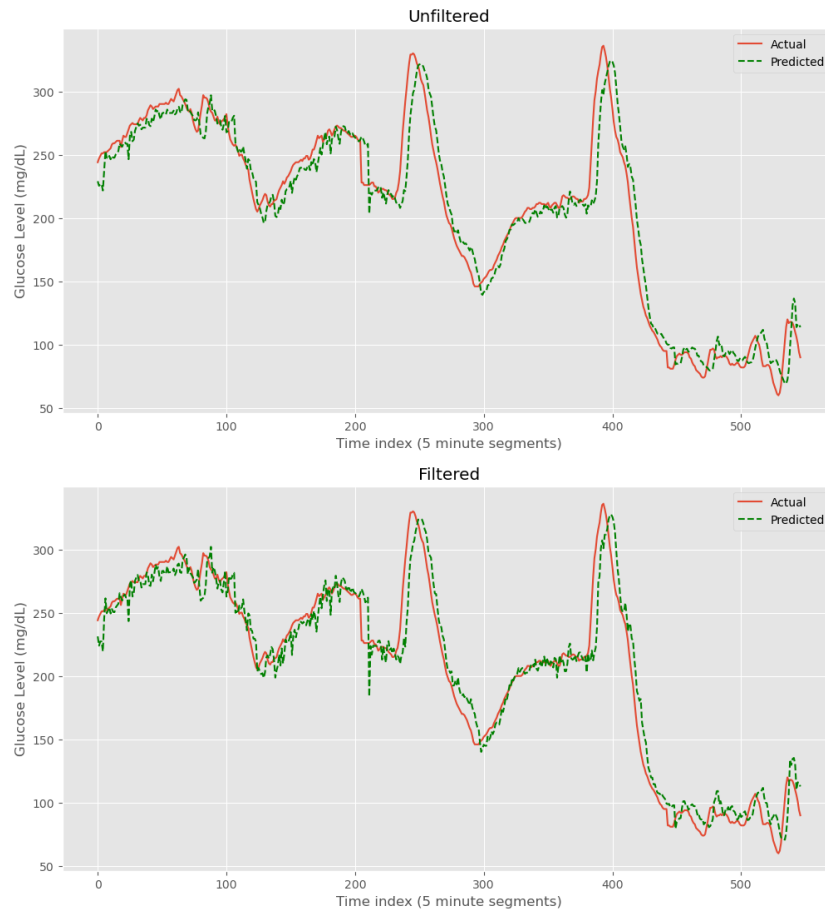
Fig. 25. Graphs showing predictions made on a segment patient 570's data at a 30 min PH

Over this segment, the performance across the two separate datasets is very similar at 17.95 / 17.58 mg/dL for the unfiltered / filtered data respectively. However, what is immediately clear is how the predictions produced by the unfiltered data produce a much smoother predicted data stream - there are far more micro peaks in the predictions on the filtered data. This is because of to the overfitting on the filtered data leading to much sharper predictions that are overspecialised to the training dataset.
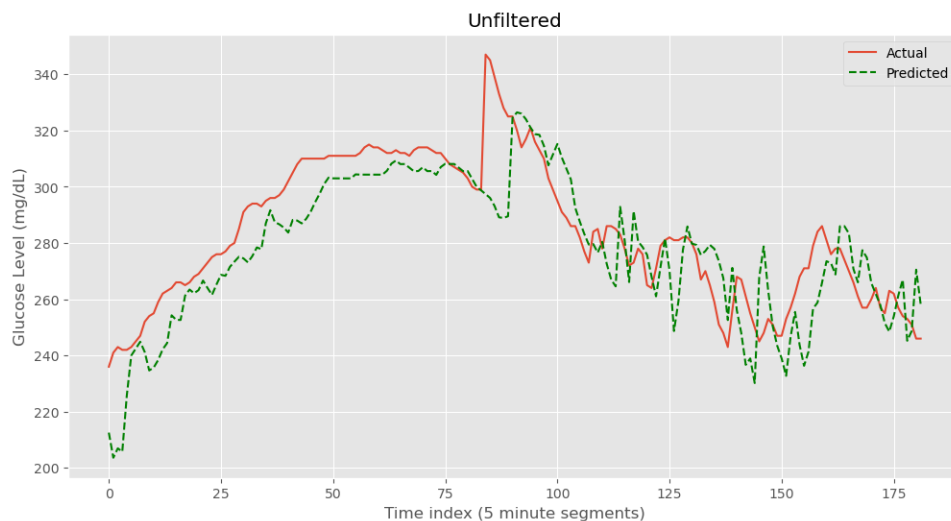

Fig. 26. A graph showing the prediction lag at a 30 minute PH for patient 570 with the basic LSTM

36

By zooming further into a separate section of patient 570's data, we can see another issue with the current LSTM model - there is a large delay between peaks and troughs in the real data and the predicted data. As the models currently only have CGM data to predict off of, sharp peaks / troughs are incredibly difficult to readily predict as there is often no inidicator in the CGM data that the BG level is about to rise sharply. Instead this indicator would more likely be within the meal or insulin data in the form of food being eaten or insulin being dosed.

Whilst both models manage to outperform the baseline in most cases, the performances are quite close which shows that these models are not performing hugely well. Multi-input models were experimented with next, in the hopes of greatly outperfoming the baseline.

4.3.3 Compartmental LSTM

Using models trained on the CGM data as well as the compartmental curves, the above experiments are repeated. To measure the performance improvement from normalisation on these curves, the model is trained both on data that has and that hasn't been normalised.

Table 5. A table showing the RMSE performance of the LSTM model trained with compartmental data

Unnormalised Data								
Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
12.58	20.22	31.60	10.96	18.05	28.97	10.74	19.10	29.84
Normalised Data								
Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
10.39	16.94	26.22	9.63	16.79	24.97	10.47	17.40	25.49

The inclusion of the compartmental models alone, without normalisation, improves prediction accuracy slightly for all patients but patient 563. However, when trained on the normalised data, the performance of the model improves significantly for every single patient - due to all of the input variables now being on the same scale. From this data, it is easy to say that the model trained on normalised data greatly outperforms the baseline model.
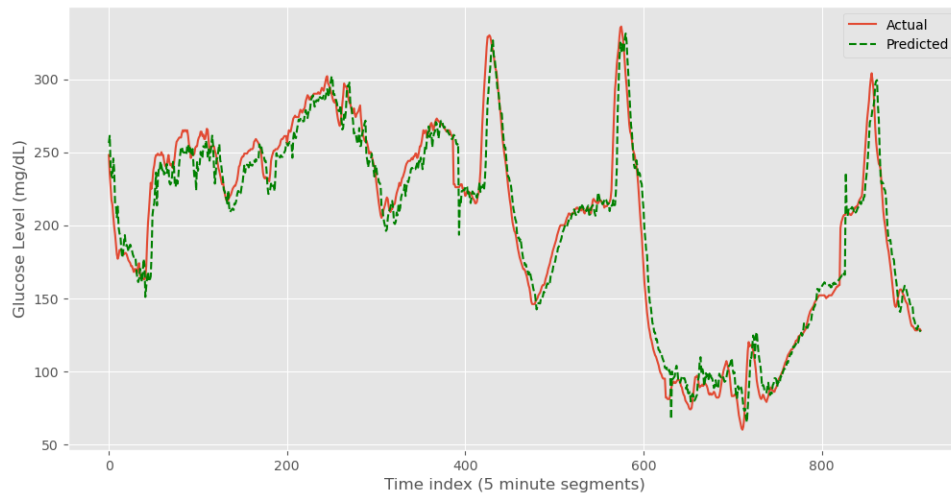
Fig. 27. A view of the predictions made by the model on normalised data for a 30 min PH on a section of patient 570's data

From this graph, the improvement in prediction is clearly visible as the green dotted line much more tightly hugs the actual red line. It is interesting that without the normalisation, there is a much larger variance in the performance of the model between the different patients than there is when the normalisation is applied. It is believed that this is because the inclusion of the compartmental models offers explanations for more unpredictable BG behaviour. For example, if the BG level quickly drops, then sharply rises, it could be explained by readings in the insulin and meal data, but not by the previous CGM data.
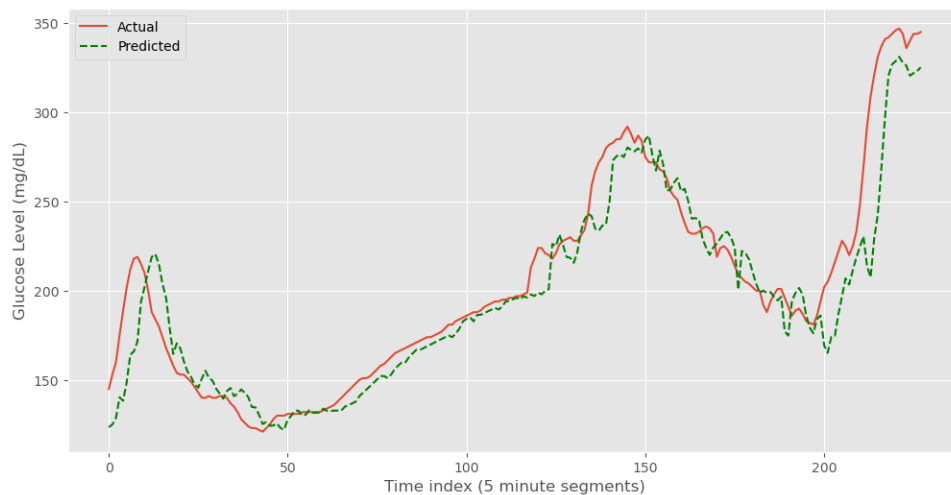
Fig. 28. A zoomed-in view of the predictions made by the model on normalised data for a 30 min PH on a section of patient 570's data

Looking at the zoomed-in view of the predictions made, it is unfortunate to see that although it exists to a slightly lesser extent, there is still a prediction delay between the actual and predicted curves. This slightly lesser delay is most likely due to datapoints in the compartmental curves giving the model some indication as to when peaks/troughs were going to occur.

As the basal data graph is not the most informative curve, and often consists of straight lines for long periods of time, a model is also trained only using insulin data and meal data to see if performance is improved.

Table 6. A table showing the RMSE performance of the LSTM model trained with no basal data

Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
10.50	16.77	25.64	9.62	16.64	25.24	9.67	16.55	23.95

Comparing these results to the results when using basal data, we can see slight improvement almost across the board except for at 60 min PH with patient 570, where we see a slight deterioration in performance. Patient 588 especially sees large improvements to performance when removing the basal curve, which likely indicates a low quality of basal data that was pulling the model's predictions in the wrong direction. This once again highlights how reliant making BG prediction models is on the given patient's data, and how their performance can vary so greatly from patient to patient.

4.3.4   Bidirectional LSTM

Finally, a Bidirectional LSTM is trained both using only CGM data, and using all of the compartmental data except for the basal data. This is done to see if a more complex model can better fit to the data and make more accurate predictions than a simple LSTM.

Table 7. A table showing the RMSE performance of the LSTM model trained with compartmental data

Just CGM data								
Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
13.01	19.71	31.47	10.30	18.69	29.69	11.75	20.11	33.21
Compartmental Data								
Patient 563			Patient 570			Patient 588		
15 min	30 min	60 min	15 min	30 min	60 min	15 min	30 min	60 min
10.76	16.92	25.72	9.49	18.60	25.85	10.24	16.31	24.03

Unfortunately, the use of a more complex, bidirectional LSTM hasn't resulted in much of an improvement to performance at all. It performs very similarly to the ordinary LSTM both when using

and when not using the compartmental data. This could be because of the fact that future BG levels are much more dependent on previous BG levels then previous are dependent on future. This fact means that the 'extra direction' of information that the bidirectional LSTM provides isn't actually very informative, and doesn't provide any noticeable improvement to model performance.

4.4 Summary

Across all of the patients, any models that were trained on both the CGM data and the compartmental data performed noticeably better. This performance was enhanced significantly further when min-max scaling normalisation [46] was applied across the input data before training. The best performing model was the simple LSTM trained on all of the normalised compartmental data apart from the basal data, with an average RMSE at 30 mins of 16.65mg/dL across the three patients. This best model significantly outperforms the baseline, whose average across the three patients was 20.77mg/dL. Patient 563 was consistently the worst performing patient across the models, with patients 570 and 588 alternating between having better performance depending on the model in question. Patient 588 saw by far the most performance increase when adding compartmental data and removing the basal data, which may have something to do with the fact that they are the only female patient, but is more likely to do with the quality of data recorded. Graphs for the performance of all of the models can be seen on the following page.
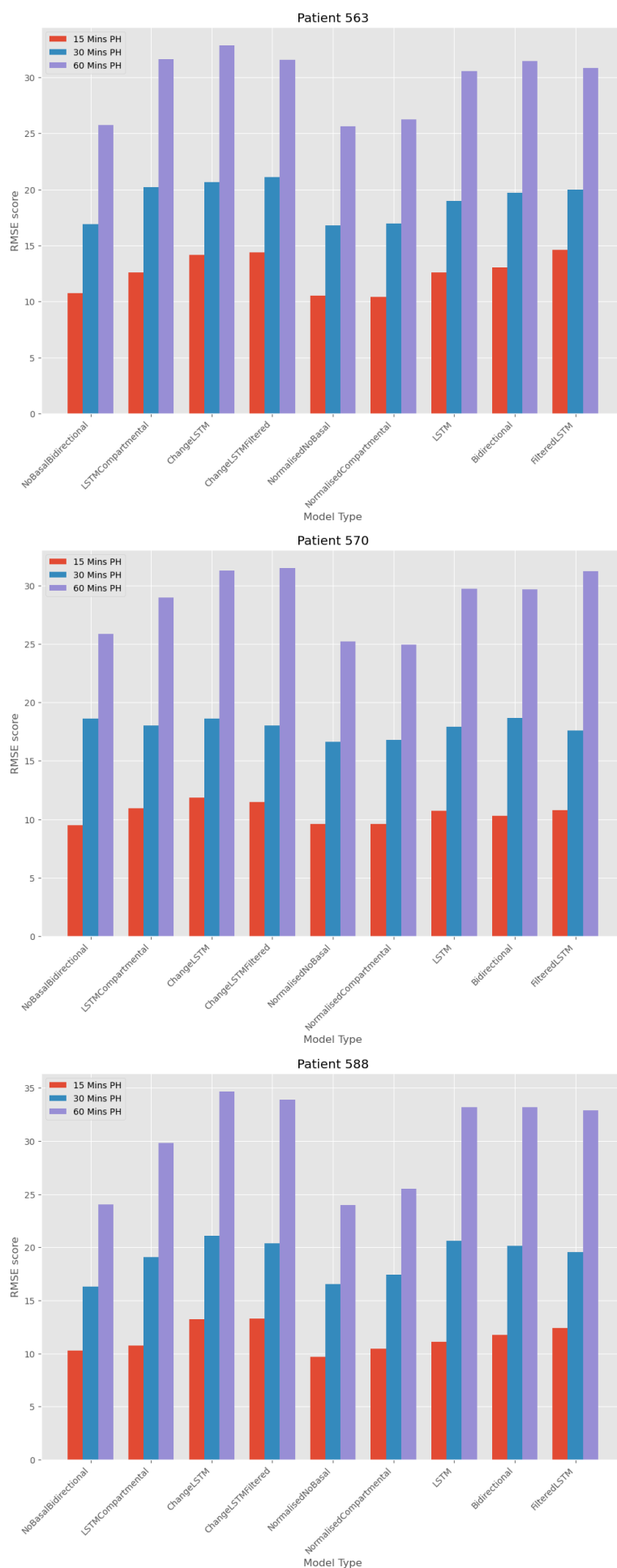
Fig. 29. Graphs showing the performance of every model across the different PHs for each patient

We also compare the performance of the best performing model with the performances seen from different approaches in literature.

Table 8. A table showing individual information about the selected patients

Author	Model Type	Accuracy (RMSE)	PH
Pérez-Gandía et al. [19]	ANN on CGM data	18.00	30 min
Alfian et al. [21]	ANN with time-domain attributes	6.31	30 min
Munoz-Organero et al. [31]	RNN on AIDA data	2.70	15 min
Robertson et al. [26]	RNN with Deep CMs	6.42	30 min
Wang et al. [33]	VMD with IPSO	3.03	30 min
Aliberti et al. [35]	LSTM multiple patients	7.18	30 min
Zhu et al. [36]	CNN on CGM data	21.72	30 min
Wang et al. [39]	LightGBM	13.90	30 min
Our model	LSTM with CMs	16.65	30 min

Our model is outperformed by most of the approaches seen in literature, however it is very hard to compare performance between them as completely different datasets are used. The simplicity of the dataset being predicted over plays a large factor in the prediction accuracy of the models. The model trained by Zhu et al. [36] was also trained on the OhioT1DM dataset, so can be directly compared, and our model makes significantly more accurate predictions.

4.4.1   Clinical Suitability

To properly evaluate the performance of a model in this context, it is important to look at the model in a clinical light. To do so, we make use of the CEG [22] to view the percentage of predictions which are clinically acceptable, that is to say the percentage of predictions in Zone A of the grid. Three models are selected for clinical experimentation:

- Basic LSTM

- LSTM with compartmental data

- LSTM with normalised compartmental data without basal

The three models are tested at all of the 5 minute PH intervals between 5 minutes and 60 minutes inclusive, to see the degradation of RMSE performance, and the percentage of predictions that remain in Zone A of the CEG.
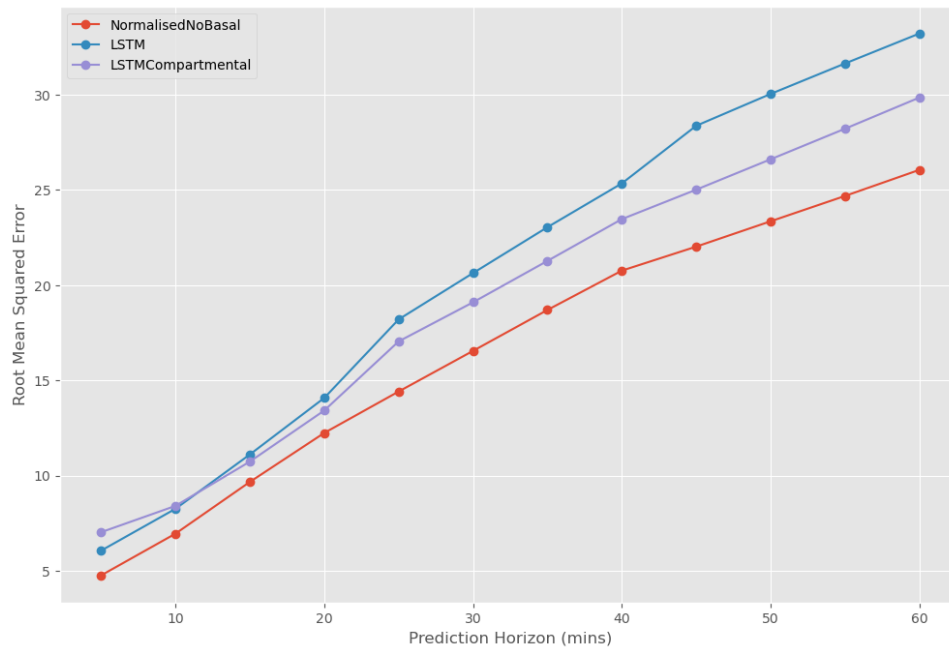
Fig. 30. A

5 Conclusions and Future Work

5.1 Conclusions

Talk generally about the parallels and differences seen from my approach vs the approaches studied in the literature review. Explain limitations of the task in reality.

5.2 Future work

Go into detail about how the model could be improved / what it would take to put such a model out into the real world via CGM/Pump.

References

[1] L. A. DiMeglio, C. Evans-Molina, and R. A. Oram, "Type 1 diabetes," *The Lancet*, vol. 391, no. 10138, pp. 2449–2462, Jun. 2018. DOI: `https : / / doi . org / 10 . 1016 / s0140 - 6736(18)31320 - 5`. [Online]. Available: `https : / / www . thelancet . com / journals / lancet/article/PIIS0140-6736(18)31320-5/fulltext` (cited on p. 6).

[2] CDC, *Insulin resistance and diabetes*, Jun. 2022. [Online]. Available: `https://www.cdc.gov/ diabetes/basics/insulin-resistance.html` (cited on p. 6).

[3] 2017. [Online]. Available: `https://www.diabetes.org.uk/diabetes-the-basics/types-of-diabetes/type-1` (cited on p. 6).

[4] M. E. Pauley, K. L. Tommerdahl, J. K. Snell-Bergeon, and G. P. Forlenza, "Continuous glucose monitor, insulin pump, and automated insulin delivery therapies for type 1 diabetes: An update on potential for cardiovascular benefits," *Current Cardiology Reports*, vol. 24, no. 12, pp. 2043–2056, Oct. 2022. DOI: `https://doi.org/10.1007/s11886-022-01799-x`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9589770/` (cited on p. 6).

[5] (2023), [Online]. Available: `https://www.who.int/data/gho/indicator-metadata-registry/imr-details/2380` (cited on pp. 8, 13).

[6] (2022), [Online]. Available: `https://www.mayoclinic.org/diseases-conditions/hyperglycemia/symptoms-causes/syc-20373631` (cited on p. 8).

[7] (2023), [Online]. Available: `https://diabetes.org/healthy-living/medication-treatments/blood-glucose-testing-and-control/hypoglycemia` (cited on p. 8).

[8] World. (May 2019). Diabetes, [Online]. Available: `https://www.who.int/health-topics/diabetes#tab=tab_1` (cited on p. 8).

[9] *Clinical Diabetes*, vol. 36, no. 2, pp. 202–202, Apr. 2018. DOI: `https://doi.org/10.2337/cd18-0012`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5898168/` (cited on p. 8).

[10] D. and. (Apr. 2023). Artificial pancreas, [Online]. Available: `https://www.niddk.nih.gov/health-information/diabetes/overview/managing-diabetes/artificial-pancreas` (cited on p. 8).

[11] J. Kesavadev, B. Saboo, M. B. Krishna, and G. Krishnan, "Evolution of insulin delivery devices: From syringes, pens, and pumps to diy artificial pancreas," *Diabetes Therapy*, vol. 11, no. 6, pp. 1251–1269, May 2020. DOI: `https://doi.org/10.1007/s13300-020-00831-z`. [Online]. Available: `https://link.springer.com/article/10.1007/s13300-020-00831-z` (cited on p. 8).

[12] Abidika, *Syringe pen and glucometer*, 2023. [Online]. Available: `https://www.dreamstime.com/stock-photo-syringe-pen-glucometer-insulin-blood-glucose-meter-showing-normal-blood-glucose-image41249489` (cited on p. 9).

[13] Sep. 2016. [Online]. Available: `https://www.umassmed.edu/dcoe/diabetes-education/pumps_and_cgm/` (cited on p. 9).

[14] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electronic Markets*, vol. 31, no. 3, pp. 685–695, Apr. 2021. DOI: `https://doi.org/10.1007/s12525-021-00475-2`. [Online]. Available: `https://link.springer.com/article/10.1007/s12525-021-00475-2` (cited on p. 10).

[15] E. Grossi and M. Buscema, *Introduction to artificial neural networks*, 2008. [Online]. Available: `https://www.researchgate.net/publication/5847739_Introduction_to_artificial_neural_networks` (cited on p. 10).

[16] 2022. [Online]. Available: `https://ml-cheatsheet.readthedocs.io/en/latest/nn_concepts.html` (cited on p. 11).

[17] Ali and H. A. Ahmed, *Gradient descent algorithm: Case study*, Dec. 2021. [Online]. Available: `https://www.researchgate.net/publication/356878797_Gradient_Descent_Algorithm_Case_Study#:~:text=Gradient%20descent%20is%20one%20of,logistic%20regression%2C%20etc...` (cited on p. 11).

[18] Crypto1, *How does the gradient descent algorithm work in machine learning?* Oct. 2020. [Online]. Available: `https://www.analyticsvidhya.com/blog/2020/10/how-does-the-gradient-descent-algorithm-work-in-machine-learning/` (cited on p. 11).

[19] C. Pérez-Gandía, A. Facchinetti, G. Sparacino, C. Cobelli, E. Gómez, M. Rigla, A. de Leiva, and M. Hernando, "Artificial neural network algorithm for online glucose prediction from continuous glucose monitoring," *Diabetes Technology Therapeutics*, vol. 12, no. 1, pp. 81–88, Jan. 2010. DOI: `https://doi.org/10.1089/dia.2009.0076`. [Online]. Available: `https://pubmed.ncbi.nlm.nih.gov/20082589/` (cited on pp. 12, 42).

[20] G. Freckmann, S. Pleus, M. Grady, S. Setford, and B. Levy, "Measures of accuracy for continuous glucose monitoring and blood glucose monitoring devices," *Journal of Diabetes Science and Technology*, vol. 13, no. 3, pp. 575–583, Nov. 2018. DOI: `https://doi.org/10.1177/1932296818812062`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6501529/#:~:text=At%20least%2095%20%25%20of%20measured,concentrations%20%E2%A9%BE100%20mg%2Fdl`. (cited on p. 12).

[21] G. Alfian, M. Syafrudin, M. Anshari, F. Benes, F. T. D. Atmaji, I. Fahrurrozi, A. F. Hidayatullah, and J. Rhee, "Blood glucose prediction model for type 1 diabetes based on artificial neural network with time-domain features," *Biocybernetics and Biomedical Engineering*, vol. 40, no. 4, pp. 1586–1599, Oct. 2020. DOI: `https://doi.org/10.1016/j.bbe.2020.10.004`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0208521620301248` (cited on pp. 13, 42).

[22] W. Contributors, *Clarke error grid*, Apr. 2022. [Online]. Available: `https://en.wikipedia.org/wiki/Clarke_Error_Grid` (cited on pp. 13, 17, 42).

[23] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132 306, Mar. 2020. DOI: `https://doi.org/10.1016/j.physd.2019.132306`. [Online]. Available: `https://arxiv.org/abs/1808.03314` (cited on p. 13).

[24] 2015. [Online]. Available: `https://colah.github.io/posts/2015-08-Understanding-LSTMs/` (cited on p. 14).

[25] S. Saxena, *Learn about long short-term memory (lstm) algorithms*, Mar. 2021. [Online]. Available: `https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/` (cited on p. 14).

[26] G. Robertson, E. D. Lehmann, W. Sandham, and D. Hamilton, "Blood glucose prediction using artificial neural networks trained with the aida diabetes simulator: A proof-of-concept pilot study," *Journal of Electrical and Computer Engineering*, vol. 2011, pp. 1–11, 2011. DOI: `https://doi.org/10.1155/2011/681786`. [Online]. Available: `https://www.hindawi.com/journals/jece/2011/681786/` (cited on pp. 14, 42).

[27] F. Alloatti, A. Bosca, L. Di Caro, and F. Pieraccini, "Diabetes and conversational agents: The aida project case study," *Discover Artificial Intelligence*, vol. 1, no. 1, Sep. 2021. DOI: `https://doi.org/10.1007/s44163-021-00005-1`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8456073/#:~:text=The%20Artificial%20Intelligence%20Diabetes%20Assistant,a%20speech%2Dbased%20dialog%20system.` (cited on p. 14).

[28] D. A. Rubenstein, W. Yin, and M. D. Frame, "Mass transport and heat transfer in the microcirculation," *Biofluid Mechanics*, pp. 331–374, 2022. DOI: `https://doi.org/10.1016/b978-0-12-818034-1.00008-6`. [Online]. Available: `https://www.sciencedirect.com/topics/engineering/compartmental-modeling` (cited on p. 15).

[29] ——, "Mass transport and heat transfer in the microcirculation," *Biofluid Mechanics*, pp. 331–374, 2022. DOI: `https://doi.org/10.1016/b978-0-12-818034-1.00008-6`. [Online]. Available: `https://www.sciencedirect.com/topics/engineering/compartmental-modeling` (cited on p. 15).

[30] S. Mougiakakou, K. Prountzou, and K. Nikita, "A real time simulation model of glucose-insulin metabolism for type 1 diabetes patients," *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, 2005. DOI: `https://doi.org/10.1109/iembs.2005.1616403`. [Online]. Available: `https://pubmed.ncbi.nlm.nih.gov/17282172/` (cited on pp. 15, 16).

[31] M. Munoz-Organero, "Deep physiological model for blood glucose prediction in t1dm patients," *Sensors*, vol. 20, no. 14, p. 3896, Jul. 2020. DOI: `https://doi.org/10.3390/s20143896`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7412558/` (cited on pp. 16, 42).

[32] F. Dubosson, J.-E. Ranvier, S. Bromuri, J.-P. Calbimonte, J. Ruiz, and M. Schumacher, "The open d1namo dataset: A multi-modal dataset for research on non-invasive type 1 diabetes management," *Informatics in Medicine Unlocked*, vol. 13, pp. 92–100, 2018. DOI: `https://doi.org/10.1016/j.imu.2018.09.003`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S2352914818301059` (cited on p. 16).

[33] W. Wang, M. Tong, and M. Yu, "Blood glucose prediction with vmd and lstm optimized by improved particle swarm optimization," *IEEE Access*, vol. 8, pp. 217 908–217 916, 2020. DOI: `https://doi.org/10.1109/access.2020.3041355`. [Online]. Available: `https://ieeexplore.ieee.org/document/9281120` (cited on pp. 16, 17, 42).

[34]   D. C. Benesch, *Error grid analysis*, 2017. [Online]. Available: `https://blog.profil.com/blog/error-grid-analysis` (cited on p. 17).

[35]   A. Aliberti, I. Pupillo, S. Terna, E. Macii, S. Di Cataldo, E. Patti, and A. Acquaviva, "A multi-patient data-driven approach to blood glucose prediction," *IEEE Access*, vol. 7, pp. 69 311–69 325, 2019. DOI: `https://doi.org/10.1109/access.2019.2919184`. [Online]. Available: `https://ieeexplore.ieee.org/document/8723121` (cited on pp. 17, 18, 42).

[36]   T. Zhu, K. Li, P. Herrero, J. Chen, and P. Georgiou, *A Deep Learning Algorithm For Personalized Blood Glucose Prediction*. [Online]. Available: `https://ceur-ws.org/Vol-2148/paper12.pdf` (cited on pp. 18, 42).

[37]   M. C. R, "The ohiot1dm dataset for blood glucose level prediction: Update 2020," *CEUR workshop proceedings*, vol. 2675, 2020. [Online]. Available: `https://pubmed.ncbi.nlm.nih.gov/33584164/` (cited on pp. 18–20).

[38]   L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions," *Journal of Big Data*, vol. 8, no. 1, Mar. 2021. DOI: `https://doi.org/10.1186/s40537-021-00444-8`. [Online]. Available: `https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8` (cited on p. 18).

[39]   Y. Wang and T. Wang, "Application of improved lightgbm model in blood glucose prediction," *Applied Sciences*, vol. 10, no. 9, p. 3227, May 2020. DOI: `https://doi.org/10.3390/app10093227` (cited on pp. 18, 42).

[40]   G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, *LightGBM: A Highly Efficient Gradient Boosting Decision Tree*. [Online]. Available: `https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf` (cited on p. 18).

[41]   2022. [Online]. Available: `https://www.diabetes.org.uk/guide-to-diabetes/managing-your-diabetes/treating-your-diabetes/insulin/types` (cited on p. 22).

[42]   H. Wang, J. Li, and Y. kuang, "Mathematical modeling and qualitative analysis of insulin therapies," *Mathematical Biosciences*, vol. 210, no. 1, pp. 17–33, Nov. 2007. DOI: `https://doi.org/10.1016/j.mbs.2007.05.008`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0025556407001058` (cited on p. 22).

[43]   L. F. Shampine and S. Thompson, "Delay differential equations," *ResearchGate*, Jan. 2009. [Online]. Available: `https://www.researchgate.net/publication/228884162_Delay_Differential_Equations` (cited on p. 22).

[44]   N. Gallagher, "Savitzky-golay smoothing and differentiation filter," *ResearchGate*, Jan. 2020. [Online]. Available: `https://www.researchgate.net/publication/338518012_Savitzky-Golay_Smoothing_and_Differentiation_Filter` (cited on p. 25).

[45] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv.org*, 2016. [Online]. Available: `https://arxiv.org/abs/1603.07285` (cited on p. 26).

[46] S. Loukas, *Everything you need to know about min-max normalization: A python tutorial*, May 2020. [Online]. Available: `https://towardsdatascience.com/everything-you-need-to-know-about-min-max-normalization-in-python-b79592732b79` (cited on pp. 26, 40).

[47] 2023. [Online]. Available: `https://www.tensorflow.org/` (cited on p. 28).

[48] M. Alom, "Adam optimization algorithm," *ResearchGate*, Jun. 2021. [Online]. Available: `https://www.researchgate.net/publication/352497171_Adam_Optimization_Algorithm` (cited on p. 30).

[49] 2020. [Online]. Available: `https://paperswithcode.com/method/bilstm` (cited on p. 31).

[50] kriventsov, *Clarke-and-parkes-error-grids/error$_g$rids.pyatmaster·kriventsov/clarke$-$and$-$parkes$-$error$-$grids*, 2023. [Online]. Available: `https://github.com/kriventsov/Clarke-and-Parkes-Error-Grids/blob/master/error_grids.py` (cited on p. 32).

[51] L. Li, K. Jamieson, and A. Rostamizadeh, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *Journal of Machine Learning Research*, vol. 18, pp. 1–52, 2018. [Online]. Available: `https://jmlr.org/papers/volume18/16-558/16-558.pdf` (cited on p. 33).