UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

MÁSTER UNIVERSITARIO EN INGENIERÍA
BIOMÉDICA

MASTER'S THESIS

DESIGN AND IMPLEMENTATION OF A LONG-SHORT TIME MEMORY NEURAL NETWORK FOR ONLINE GLUCOSE PREDICTION

JAIME MARÍA CARRILLO MORENO

2019

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

MÁSTER UNIVERSITARIO EN INGENIERÍA BIOMÉDICA

MASTER'S THESIS

DESIGN AND IMPLEMENTATION OF A LONG-SHORT TIME MEMORY NEURAL NETWORK FOR ONLINE GLUCOSE PREDICTION

JAIME MARÍA CARRILLO MORENO

Tutor:

ÁLVARO GUTIÉRREZ MARTÍN

2019

Resumen

La diabetes afecta a alrededor de 422 millones de personas en el mundo, siendo uno de los grandes problemas de salud en la actualidad. La diabetes engloba un grupo de enfermedades que se caracteriza por tener niveles elevados de glucosa en sangre. Este proyecto se centra en la diabetes tipo 1. Las complicaciones de la diabetes están provocadas por las hiperglucemias e hipoglucemias. Estas complicaciones pueden acarrear otras enfermedades tales como infartos, accidentes cerebro-vasculares, ceguera o hasta la muerte. Sin embargo, estas complicaciones pueden evitarse o, al menos, reducirse controlando los niveles de glucosa en un rango normal.

El uso de un predictor de los futuros niveles de glucosa puede ayudar a los pacientes a planificarse y a afectuar las modificaciones necesarias a su tratamiento para evitar los eventos de hiperglucemia y de hipoglucemia. Para la implementación de este predictor se ha optado por utilizar redes neuronales artificiales, en concreto se ha elegido el uso de redes neuronales recurrentes.

En este Trabajo de Fin de Máster se ha diseñado un predictor basado en redes neuronales recurrentes, específicamente se ha usado Long-Short Time Memory (LSTM). Este predictor se ha desarrollado para pacientes con diabetes tipo 1, como parámetros de entrada al predictor se han utilizado tres de ellos: los valores pasados de glucosa proporcionados por un sensor de medición continua, las unidades de insulina proporcionadas por una bomba de insulina y, por último, la cantidad de carbohidratos ingerida. Además, se han establecido 4 tiempos de predicción: 5, 15, 30 y 45 minutos. A su vez, se han considerado 3 valores para las dimensiones de entrada: 5, 20 y 50. Para terminar, se han desarrollado 12 predictores cada uno especializado en un tiempo de predicción y en una dimensión de entrada.

Palabras claves: Redes neuronales artificiales, Long-Short Time Memory, LSTM, series temporales, predicción de glucosa, diabetes, diabetes tipo 1.

Abstract

Diabetes affects around 422 millions of people worldwide, making it a the current major healthcare problem. Diabetes is a group of diseases that is characterized by high blood sugar. This project focuses on type 1 diabetes. Diabetes complications are caused by the occurrence of hyperglycemia and hypoglycemia. These complications may produce other diseases such as heart attack, stroke, blindness or even death. Nevertheless, these complications may be avoided or, at the very least, reduced by controlling the blood glucose concentration in a normal range.

A predictor of future glucose levels may help patients plan and make necessary modifications to their treatment to avoid hyperglycemic and hypoglycemic events. This predictor was implemented using artificial neural networks, specifically, it uses recurrent neural networks.

In this Master's Thesis, a recurrent neural network based predictor that uses Long-Short Time Memory (LSTM) was designed. This predictor is specific for type 1 diabetes patients. The input parameters of the predictor are the past glucose levels by a continuous measurement sensor, the insulin units provided by a insulin pump, and, lastly, the carbohydrates intake. In addition, 4 prediction times have been established: 5, 15, 30 and 45 minutes. At the same time, 3 input dimensions have been selected: 5, 20 and 50. To conclude, twelve predictors have been developed, each one is specialized in a prediction time and an input dimension.

Key words: Artificial neural networks, Long-Short Time Memory, LSTM, temporal series, glucose prediction concentration, diabetes, type 1 diabetes.

Acknowledges

Quisiera empezar estas líneas agradeciendo a todas la personas que de una forma u otra han ayudado a que se llevase a cabo este proyecto. En primer lugar, me gustaría agradecerle a mi tutor, Dr. Álvaro Gutierrez, por confiar en mí para hacer este proyecto y por todo el apoyo prestado. También a los compañeros de Robolado, en concreto a Rafa por todo el apoyo prestado durante la realización de este Trabajo de Fin de Máster.

También quisiera agradecer a mi familia por creer en mí en todo momento, especialmente a mis padres que con su ayuda y comprensión han permitido que pudiera llevar a cabo este proyecto, y a mi hermano Pablo por sus innumerables consejos y su apoyo en todo momento. De la misma manera, agradecer a Javier por su generosidad y por su constante ayuda, asimismo, me gustaría agradecer a Morgan porque su apoyo y confianza durante todo este tiempo han sido fundamentales para que pudiese acabar este trabajo. Por último, me gustaría recordar a aquellos que ya no están pero que sin los cuales no estaría donde estoy hoy.

Contents

List of Figures

List of Tables

Acronyms

ARM: Auto-Regressive Model

BPTT: Back-Propagation Through Time

CGM: Continuous Glucose Monitoring

CBGM: Capillary Blood Glucose Measurement

CNN: Convolutional Neural Network

CUDA: Compute Unified Device Architecture

DM: Diabetes Mellitus

FIR: Finite Impulse Response

GBT: Grupo de Bioingeniería y Telemedicina

GPU: Graphical Processing Units

ID: Input Dimension

LSTM: Long Short Term Memory

MSE: Mean Square Error

NNM: Neural Network Models

NRMSE: Normalized Root Mean Square Error

RMSE: Root Mean Square Error

RNN: Recurrent Neural Network

RTRL: Real-Time Recurrent Learning

PDA: Personal Digital Assistant

PT: Prediction Time

SGD: Stochastic Gradient Descent

WHO: World Health Organization

Chapter 1

Introduction

1.1 Context

Diabetes mellitus (DM), also known as diabetes, is a group of diseases characterized by high blood sugar levels over a long period of time [17]. This disease is highly related to insulin, a hormone that regulates blood sugar, also referred to as glucose, in the body. The underlying cause of diabetes varies from each of the four types: type 1 diabetes, type 2 diabetes, gestational diabetes and other specific types of diabetes [1].

Diabetes is an important health care problem. According to a global report by the World Health Organization (WHO) [51], around 422 million adults, globally, had diabetes in 2014. The diseases prevalence and the number of cases of diabetes continues to increase. Diabetes was found to be the direct cause in 1.5 million deaths in 2012. Additionally, in the same year, 2.2 million deaths were due to complications of the disease. Possible complications caused by diabetes are heart attacks, strokes, kidney failure, etc [65]. These complications are some of the main costs in the healthcare system and could be reduced with better control of diabetes [15].

The treatment for this disease is blood glucose control and with it, complications can be prevented. The objective of blood glucose control is to minimize the occurrences of hypoglycemia and hyperglycemia, low and high blood glucose levels, respectively. Normoglycemia is the normal concentration of blood glucose, this is achieved when the values of glucose are between 80–130 mg/dl for the fasting plasma glucose, and less than 180 mg/dl for one or two hours after a meal for the postprandial plasma glucose [58]. Factors that influence the blood glucose levels are diverse, including, but not limited to, diet, physical exercise, insulin therapy, ... These factors can be addressed and are often main treatments to combat diabetes.

To track and control blood sugar levels, self-monitoring by the patient is necessary [9]. In order to do this, different types of glucometers are available in the market. The traditional models require that the patient prick themselves to get a blood sample (capillary blood glucose measurements), however, this method can be painful [21]. Therefore, the capillary blood glucose measurements (CBGMs) are often not monitored continuously throughout the day, resulting in only a few samples being taken daily. Currently, there are also systems that provide a continuous glucose monitoring (CGM) obtaining subcutaneous glucose measurements every few minutes.

CGM sensors, such as the Medtronic [Northridge, CA] Guardian® [38], are

minimally invasive systems that measure the glucose in the interstitial fluid instead of the plasma. The measurement of glucose in the interstitial fluid is delayed with respect to the blood glucose levels, this delay could be longer than 10 minutes [19]. The use of a glucose predictor could provide valuable information, making the data coming from the CGM in a time horizon more useful. Additionally, a predictor would allow a patient to plan their actions, and therefore, deal with the disease. Although, to have a useful predictor, two factors have to be taken into account. First, the time prediction horizon has to be long enough so patients can adjust their actions. Secondly, according to [49] the fact that there is a delay between the CGM system measurement and the blood glucose value, means it is important that the time prediction horizon is longer than the delay so the patient can anticipate any changes that may occur in his or her blood glucose levels.

Different techniques have been followed to predict the blood glucose level as, for instance, neural network models (NNMs), auto-regressive models (ARMs), recurrent neural networks (RNNs), convolutional neural networks (CNNs).

[49] proposed a NNM to predict glucose in real time from two different sensors. The model only used CGM measurements as input data. The algorithm provided three different prediction times for 15, 30 and 45 minutes. Moreover, they compared their model with the [62] ARM model, obtaining better results using the NNM. This proved that the use of artificial neural networks could fit in this kind of problem.

In addition to previously mentioned approaches, [68] proposed a model that was built from CNN layers and employed WaveNet algorithms. They used the database OhioT1DM dataset[1] developed by [35] that contains type 1 diabetes patients' glucose levels, insulin bolus, carbohydrate intake and time. They also used other input data such as heart rate and temperature. However, these new parameters hindered the performance of their model.

A RNN, proposed by [2], used input data from the previous values of a CGM sensor. They compared their model and the model of [49]. The results showed that the RNN architecture was better in making predictions than the NNM model.

Neural network techniques have been used in the past and having good performance in the prediction of blood sugar levels. The use of RNN is recommended because their feedback connections allow them to learn temporal dependencies [2]. This Master's Thesis presents a model using long short term memory neural networks (LSTM). LSTMs are a type of RNNs designed by [23].

1.2 Motivation and objectives

The main goal of this Thesis is to design and implement a set of predictors with the aim of predicting the glucose level of type 1 diabetics. First, a main predictor model developed will consist of LSTM artificial neural networks fed with previous values of glucose, insulin bolus and meal intake. Based in the main predictor model, a set of predictors specialized in forecasting for different prediction times (PTs) will be deployed. Furthermore, the use of different input dimensions in the performance of the predictors will be explored. The Master's Thesis is based and extends the work

[1] Avalaible in: http://smarthealth.cs.ohio.edu/OhioT1DM-dataset.html

of [49].

To achieve the objectives described above, this Thesis will complete the following:

- Study diabetes disorder, focusing on diabetes type 1.
- Explore the basis of recurrent neural networks, and specially LSTMs neural network and their use in forecasting.
- Design a predictor based on a LSTM architecture.
- Train the predictor to find the optimal parameters for each PT (5, 15, 30 and 45 minutes) and for each input dimension (5, 20 and 50).
- Validate the set of models specialized in predictions for each PT and each input dimension.

1.3 Document layout

This Master's Thesis is divided into six chapters:

- Chapter 1 presents the objectives, the motivation and the most important aspects of each chapter.
- Chapter 2 will explain the function of insulin and glucagon and how these two hormones helps to control the blood sugar levels. It will continue explaining the main types of diabetes mellitus, and which is the effect of the hypoglycemia and hyperglycemias. Lastly, it will present the main treatments for diabetes type 1 and the different devices used to control the glucose level.
- Chapter 3 will start by presenting the different paradigms of learning for artificial neural networks (ANNs). Then, the perceptron the most basic ANN will be presented. In addition, it will provide insight of the RNNs. It will also introduce the back-propagation through time (BPTT) algorithm, an algorithm to perform the training of the RNN. Moreover, it will present the limitations of the RNN and the problem of the vanishing and exploding gradient. To conclude, it will give an introduction to the basics concepts of LSTM neural network and the main reasons of its creation.
- Chapter 4 will start by presenting the development environment. Then the chapter will continue by explaining the nature of the dataset and the pre-processing techniques used to prepare the data to feed the proposed predictor. Furthermore, it will show the process of arrange the data for the training and evaluation for the proposed predictor. In addition, the main architecture of the LSTM based predictor model will be presented. Finally, it will show the search of the optimal parameters to predict for each PT and each input dimension, concluding with a set of predictors optimized for each PT and each input dimension.

- Chapter 5 will be dedicated to assess the set of predictors designed in Chapter 4. First to be presented will be the metrics used to measure the performance of the models. Afterwards, the results from the different studies carried out in this chapter will be presented. To conclude, the results will be discussed.
- Chapter 6 will give the conclusions of the Thesis and outline possibilities of future research.

Chapter 2

Background

This Chapter aims to give a brief review of the current knowledge about diabetes, focusing on type 1 diabetes. First, it will give an introduction of the importance of the main hormones that control the blood sugar levels: insulin and glucagon. It will continue by classifying the different types of diabetes, the main consequences of the diseases, and the main methods of treatment for type 1 diabetes. To conclude, it will present some of the technology used to control diabetes. The main ideas exposed in this Chapter comes from [26], [6] and [48].

2.1 Mechanism of blood sugar control: Insulin and Glucagon

There are two main hormones, insulin and glucagon, that help to control blood sugar levels in the body. Insulin is a hormone that is secreted in the pancreas, specifically, in the beta (β) cells of the islets of Langerhans. One of the main functions of insulin is to decrease the blood glucose level. In the alpha (α) cells of the islets of Langerhans, glucagon is secreted. Its main function is to raise the blood glucose concentration, therefore, it has the opposite effect of insulin.

The insulin levels of a person who is fasting are between 5 and 25 μ/ml [14]. There are different factors that increase these levels. The secretion of insulin depends mainly on the blood sugar level, for instance, after the absorption of the carbohydrates there is an increase of the blood glucose levels and then the secretion of insulin grows. Other factors that increase insulin levels of secretion include the presence of some amino-acids in conjunction with the increase of the glycemia, or also known as blood glucose levels. Moreover, gastrointestinal hormones, such as gastrin and secretine, are released by the digestive tract after a meal. These hormones induce an early secretion of insulin as a preventive action to the absorption of glucose after meals.

Glucagon and growth hormone are hormones that stimulate the secretion of insulin. On one hand, hormones, such as Glucagon, may increase the secretion of insulin indirectly by the strengthening of the glucose stimulus. However, during hypoglycemic events, a stimulation of the sympathetic nervous system increases the secretion of glucagon is produced and, at the same time, the secretion of insulin is inhibited. On the other hand, other hormones, such as growth hormone, directly stimulate the secretion of insulin.

The stimulation of the parasympathetic nerves also increases secretion of insulin during a hyperglycemia event.

Insulin impacts glucose in two ways that result in the reduction of blood glucose levels. First, insulin promotes the glucose uptake by the cells. Glucose needs the action of insulin to pass though the membrane cell and this way can be used for the cells as a source of energy. There are exceptions, for instance, the brain does not need insulin to be able to use the glucose, because brain cells are permeable to glucose and can use glucose without the insulin intervention. Glucose is the main source of energy for the brain, which needs a continuous input of glucose [10]. If glucose levels decrease too much, an individual may go into coma. Other tissues that do not need the action of insulin to use glucose are the retinas and the kidneys.

The second role of insulin consists of the storage of the glucose for later use. There are two methods in which the glucose is stored. The insulin promotes the glycogenesis, which is the process that stores the glucose in the form of glycogen. This glycogen is stored in the liver to be used later in order to maintain normal blood sugar levels in times of fasting. Glycogen is also stored in muscular cells to be used during times of physical activity. Second method happens when the blood glucose levels are so high, that all the glucose cannot be stored as glycogen, then insulin promotes the conversion of glucose into fatty acids that are later converted and deposited as fats.

The deficit of insulin promotes the use of fats as an energy source, which produces Acetyl Co-enzyme A. If this deficit is prolonged over a long period of time then the Acetyl Co-enzyme A is used to produce ketone bodies. These ketone bodies modify the pH of the blood and may cause ketoacidosis and possibly a coma that may end in death. Moreover, the deficit of insulin decreases the number of proteins because there is an increase of the catabolism of proteins and also the protein's synthesis stops. All this contributes to an increase of amino-acids in the blood plasma. These amino-acids are used as a source of energy or as substrate for the gluconeogenesis, the formation of glucose from substances other than carbohydrates. The loss of proteins can produce an alteration of several functions of the organs.

Glucagon is the other hormone that helps to maintain the blood glucose levels between the normal values. Its main function is opposite to the one of the insulin. Glucagon raises the levels of blood sugar in one of two ways that glucagon promotes: glycogenolysis and gluconeogenesis. The glycogenolysis is the extraction of glucose from the glycogen in the liver and is the main source of glucose for short fasting periods. In addition, the glucagon increases the gluconeogenesis in the liver and if the fasting last for a long period of time and glycogen reserves are depleted, the gluconeogenesis is the only source of glucose [54].

2.2 Diabetes mellitus

Diabetes mellitus, also known as diabetes, is a major healthcare problem [63]. According to the global report by the World Health Organization (WHO) [51], around 422 million adults had diabetes in 2014 and the number of cases and the prevalence are increasing. Diabetes was the direct caused in 1.5 million deaths and 2.2 million deaths were due to complications of the disease in 2012. Possible complications are heart attacks, strokes, kidneys failure, among others [65]. These complications are

some of the main costs in the healthcare system. These costs could be avoided with a better control of diabetes. [15].

Diabetes is a chronic disease that occurs when the pancreas does not produce enough insulin or when the body has a resistance to the insulin that it produces [51]. Insulin is essential for glucose to function properly in the metabolism, as can be seen in Section 2.1.

2.2.1 Types of diabetes

There are different types of diabetes and the underlying cause of diabetes varies from type to type. Diabetes is classified in four different groups: type 1 diabetes, type 2 diabetes, gestational diabetes and other specific types of diabetes [1].

2.2.1.1 Type 1 diabetes

Type 1 diabetes, also known as juvenile diabetes or insulin-dependent diabetes, is a chronic condition. The usual onset age of type 1 diabetes is puberty, but can start at any age [17].

Type 1 diabetes can be caused by injury of β cells in the pancreas or by diseases that impair insulin production. The causes of the destruction to β cells include, but are not limited to, autoimmune diseases and viral infections. Moreover, genetic inheritance and environmental factors also play an important role in the degeneration of β cells [26].

The main symptoms that patients can suffer are related to the high levels of blood sugar. These symptoms include, but are not limited to: dry mouth, constant urination, extreme fatigue and quick weight loss, etc [17].

2.2.1.2 Type 2 diabetes

Type 2 diabetes is the most frequent case of diabetes. According to [17], around 90 % of all cases of diabetes are type 2 diabetes. In most cases, type 2 diabetes is seen in adults older than 50 years old, although it seems to be an increase in the number of cases in younger ages. The causes of type 2 diabetes are not completely understood, however, it has a strong relation with obesity, sedentary lifestyle, age and genetic inheritance [17].

Type 2 diabetes is due to an insulin resistance in the patient's cells in addition to the pancreas not being able to make enough insulin to overcome this resistance [26]. The symptoms may be identical to the symptoms on the type 1 diabetes, although, the onset of the type 2 diabetes is slower and patients can live for long periods of times without the acute symptoms of the disease[17].

The most common treatment of type 2 diabetes is exercise, diet control, weight reduction and pharmacological treatment. This all helps to control the disease without insulin injections [26]. This treatment helps to remit or improve the patient condition.

2.2.1.3 Gestational diabetes

According to [17] Gestational diabetes is a temporary condition that occurs in pregnancy and it resolves when pregnancy ends. However, it can also carry the

long-term risk of type 2 diabetes. Moreover, after birth, those children also have a higher risk to develop type 2 diabetes. Gestational diabetes is usually diagnosed in the second and the third trimester, although it also can be detected in the first one [17].

2.2.1.4 Other specific types of diabetes

This last type of diabetes is made up of less common causes where the cause that leads to diabetes is identified in a particular form [1]. This group includes, but is not limited to: pancreatopathy, infections, genetic syndromes among others [65].

2.2.2 Hypoglycemia and hyperglycemia events

The normal concentration of blood glucose, normoglycemia, is achieved when the values of glucose are between 80–130 mg/dl for the fasting plasma glucose, and less than 180 mg/dl one or two hours after a meal for the postprandial plasma glucose [58]. The postprandial plasma glucose is the concentration of blood sugar after a meal. Type 1 diabetics cannot achieve these values without external help because they have a deficit of insulin production, that is what causes the hypoglycemia and hyperglycemia events.

Hypoglycemia and hyperglycemia are the two majors consequences of diabetes, and they provoke most of the complications associated with the disease [48]. Hypoglycemia happens when the blood sugar is below 70 mg/dl [4]. As stated in Section 2.1, the main source of energy for the brain is glucose. Therefore, hypoglycemia causes severe dysfunction in the nervous system, including coma and death [65].

According to [65], a hypoglycemia can occur for different reasons. One of the causes can be the use of more than the needed insulin, these could reduce the blood sugar below the normal levels. Moreover, physical activity can lead to a hypoglycemic event, in addition to fasting for long periods, such as a night sleep. Not eating a proper meal may turn out to be the cause of low blood glucose level too [65].

Hyperglycemia is a higher blood glucose level than the normoglycemia values. A value higher than 180 mg/dl two hours after a meal is considered hyperglycemia [3]. A prolonged hyperglycemia over time can cause damage in the tissues. This damage may produce macrovascular and microvascular diseases, such as atherosclerosis and retinopathy, respectively [65]. Another disease caused by hyperglycemia is ketoacidosis [48].

2.3 Management of type 1 diabetes

Currently, the most utilized therapy to manage type 1 diabetes is intensive therapy. This intensive therapy tries to maintain the blood glucose levels in the normoglycemia range and it consists of the following: control of the glucose levels, intensive insulin therapy, nutritional therapy and physical exercise [65]. Furthermore, it is essential that the patients are trained and educated about diabetes, so they can self-manage it and develop the skills necessary for diabetes self-care [6].

2.3.1 Glycemic control

The glycemic control is used to adjust the patient treatment. The hemoglobin A1C test assess the necessary changes in the patient's treatment in the long term. In the other hand, the self-monitoring of blood glucose is used to adjust the patient treatment in the short term.

The value of the hemoglobin A1C test is a percentage that represents the blood sugar levels over the last two to three months [65]. It is used by the clinician to evaluate the glycemic control in the long term and to assess the treatment [6].

The self-monitoring of blood glucose is mostly used by the patients. The patients have been trained to use this value to make the necessary adjustment to their treatment. Therefore, the value of blood glucose levels is of high importance in the treatment of diabetes type 1. The number of measurements of the patient's capillary blood glucose is assessed by a clinician. In addition to capillary blood glucose monitoring, there is also evidences that the use of continuous glucose monitoring sensors are helpful for glycemic control [28].

[6] recommends the following glycemic values of the previous tests for adults with diabetes, with the exception of pregnant women. These values can be different for each person depending of their situation and their clinician's judgment:

- Hemoglobin A1C less than 7%.
- Fasting plasma glucose: 80 - 130 mg/dl.
- Postprandial plasma glucose: less than 180 mg/dl.

2.3.2 Insulin therapy

In intensive therapy to manage diabetes, the insulin treatment applied is also an intensive insulin therapy. This insulin therapy tries to emulate the functioning of a healthy pancreas, which is accomplished using multiple insulin injections or with insulin pumps. In this therapy, there are two kinds of bolus. The basal bolus that emulates the secretion of the pancreas in fasting periods and the postprandial to control hyperglycemia after the meal.

It is important to emphasize the importance of training and education of the patient on the disease, as it allows the patient to modify the bolus in function of the meals, exercise, and other parameters that can affect their blood sugar levels.

According to [12], insulin can also be classified by its pharmacokinetics. There are three factors that help characterize insulin. First, the *onset* defines the absorption time of the insulin by the body. Secondly, the *peak* states the moment of maximum effect of the inulin. Lastly, the *duration* establishes the time that insulin remains in the patient's body. Four types of insulin are defined in relation to the pharmacokinetics: rapid-acting insulin, intermediate-acting insulin, long-acting insulin and biphasic insulin [12]. Figure 2.1 shows a simulation of the different pharmacokinetics insulin types.

As stated before, there are two types of insulin bolus. For the basal bolus, intermediate-acting insulin and long-acting insulin are used. On the other hand,

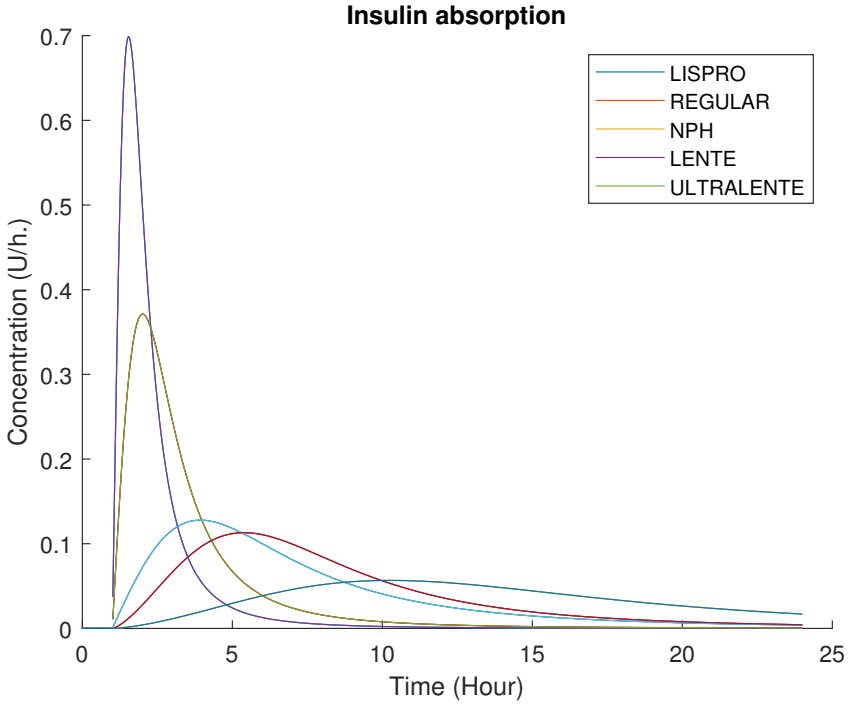


Figure 2.1: Simulation of different pharmacokinetics insulin types.

the postprandial bolus are performed with rapid-acting insulin. Biphasic insulin are used to perform both insulin bolus in only one injection [12].

2.3.3 Nutritional therapy and physical exercise

The objective of this treatment is to modify the diet to improve glycemic control [6]. The recommended diet is a balanced and healthy diet [11]. In addition, planned intake of carbohydrates improve blood glucose levels and the quality of life [34].

Physical exercise is another important part of the treatment against diabetes. It improves glycemic control because during exercise the motor cells are able to uptake glucose without the action of insulin and in this way, it reduces the blood sugar levels [26]. In addition, physical exercise increases the insulin sensitivity, so less insulin is needed to have the same effect in the patient. This should be considered in order to prevent the occurrence of a hypoglycemia. Moreover, it helps to decrease cardiovascular risk factors and helps to lose weight. It is recommended to do 150 minutes of moderate aerobic exercise per week, distributed over at least 3 to 5 days [6].

2.4 Diabetes technology

According to [5], diabetes technology refers to the devices and, also, software that helps to manage the diabetes disease. This classification is going to be divided in two main categories insulin delivery and blood glucose monitoring [5].

2.4.1 Insulin delivery

The most used methods to deliver insulin are insulin syringes and insulin pens. Insulin syringes are the traditional way to deliver insulin and consists of the syringe and the vial with the insulin. Insulin pens combine the syringe and the vial in one device. The use of insulin pens have some benefits. It is easier to use and may help people with lack of dexterity skills [45].

(a) Syringe and vial (b) Pen

Figure 2.2: (a) Insulin syringe [60] and (b) Lantus® SoloStar® insulin pen [56] systems to deliver insulin therapy.

Insulin pumps are also widely used in intensive insulin therapy. Insulin pumps try to emulate the secretion of a healthy pancreas, therefore, there are two different types of injections: basal insulin and insulin bolus. Insulin pumps only deliver one type of rapid-acting insulin. The basal insulin injections are small quantities of insulin that emulate the secretion of pancreas in the fasting periods. The insulin bolus emulates the pancreas secretion after a meal, this bolus is calculated and programmed by the patient before each meal. The use of insulin pumps improve the glycemic control and reduce the number of hypoglycemia occurrences [66].

2.4.2 Blood glucose monitoring

Individual self-monitoring of blood glucose is an important part of diabetes treatment and is even more important if the patient is using an intensive insulin therapy. To accomplish the self-monitoring of blood glucose, an important device called a glucometer is used. Glucometers are devices that measure the capillary blood glucose levels. The information provided by the blood glucose monitor helps patients make better decisions about their treatment.

Continuous glucose monitor sensors provide information about the trend, rate of change and concentration of glucose in the patient [48]. In this Master's Thesis, glucose data is provided by the Guardian® Real-Time CGM System, a CGM sensor that measures the glucose in the interstitial fluid every 5 minutes. In Section 4.2, the dataset will be furthered explained.

Figure 2.3: MiniMed™ 670G insulin pump [39].

Figure 2.4: Accu-Check® Aviva Glucometer [50].

Figure 2.5: Guardian™ Sensor 3 CGM System [40].

Chapter 3

Recurrent Neural Networks

This Chapter is a brief summary of recurrent neural networks that will be used in the predictor of this Master's Thesis. The main aspects discussed here are from [25], [16], [42] and [59].

First, a brief introduction of artificial neural networks (ANNs) and types of learning will be given.

The recurrent neural networks (RNN) will be exposed and the main ideas behind these networks. Then, the algorithm of back-propagation through time to train recurrent neural networks will be explored and the limitations of RNNs will be explained. To conclude, the long-short term memory (LSTM) neural networks, a type of RNN that were designed to overcome their limitations, will be presented.

3.1 Introduction

ANNs are inspired in a simple abstraction of the neurons of the brain [16]. These ANNs receive some inputs to generate outputs. The inputs to the ANN can be different kinds of data, for example, images, signals to be processed, time series, etc, so the ANN can produce the desired output. ANNs learn by examples, that are in training datasets, to solve a specific task. The process of learning is based on the modifications of the weights and biases in the neurons. These learning processes can be classified in three different paradigms: supervised learning, unsupervised learning and reinforcement learning ([57] [7]).

In supervised learning, the dataset used for training is labeled, so each input data has his correspondent output, also called target. Being ($I = \{i_1, i_2, ..., i_n\}$) the inputs that produce the correspondent labelled output or target ($T = \{t_1, t_2, ..., t_n\}$). An error function that measures the predictions of the ANN with the targets associated to the input data has to be defined. These error functions are propagated by the neural network model, in this way, the ANN learns from past experiences. There is also necessary a test dataset used to validate the performance of the model built with ANN. The most popular supervised learning algorithm to train ANN is the back-propagation algorithm [55].

On the other hand, in unsupervised learning the training dataset is not labeled; so the model only trains in response to its inputs. One of the most used unsupervised learning methods is the cluster analysis, which is used to find hidden patterns in the data or for clustering the data. Evaluating the performance of these models is less

straightforward than in the supervised learning. Some examples are Adjusted Rand Index, H criterion and Mutual Information, among others [64].

To conclude this classification, it is important to note the reinforcement learning rule. The model is provided with a grade for each different input value to the network. This grade or score is the performance of the model, and the model trains to maximize this score [16].

ANNs have proved successful in solving problems that are highly non-linear, where variables vary over time, or to detect patterns and trends that are complex and hidden, among others. Several examples of these solutions in different fields may be cited: in the image classification, ANN have been used for medical image analysis [33], on natural language processing ([13], [27]), and for audio signal processing [47]. The use of ANNs to predict the blood glucose levels may be a good approach as shown in Section 1.1. The forecast of blood glucose levels will be a supervised learning problem.

One of the simplest ANN is the perceptron [53], and it will be used in this Thesis as the final layer of the proposed predictor, see Section 4.3. The perceptron formulation is shown in Equation 3.1.

$$y = \varphi\left(Wx + b\right) \tag{3.1}$$

The internal parameters of neural networks are the weight matrix (W) and the vector bias (b). These internal parameters are applied to the input vector to the perceptron (x). The activation function (φ) is applied to the results of applying the internal parameters to the input vector. This produces the output vector with predictions made by the perceptron (y). The activation function may be linear or non-linear. The choice of a specific activation function is determined by the specific problem to solve, see Figure 3.1.

3.2 Recurrent Neural Networks

The choice of the type of the network architecture is strongly influenced by the type of problem to be solved. In this Master's Thesis, a forecast time-series is stated thus a LSTM architecture has been chosen as the most optimal solution. LSTM are a type of recurrent neural networks, and it would be explained further in this Chapter (see section 3.2.2). Recurrent neural networks (RNN) are designed to process sequences. They excel in processing sequential data because they have recurrences that provide memory to the RNN [25]. This memory enables the RNN to learn sequential or temporal patterns.

There are several ways to build RNNs, because most recurrent functions can be considered a RNN [25]. To show how a RNN works, a recurrent neural network is presented in Eq 3.2.

$$h_k = \varphi\left(Wx_k + Uh_{k-1} + b\right) \tag{3.2}$$

where k stands for the time step and h_k is the output of the RNN. Moreover, x_k is the input vector that is a sequence of values from $k = \{1, ..., \tau\}$, being τ a number of time steps that belongs to $\mathbb{N}$. In addition to the internal parameters W and b, the

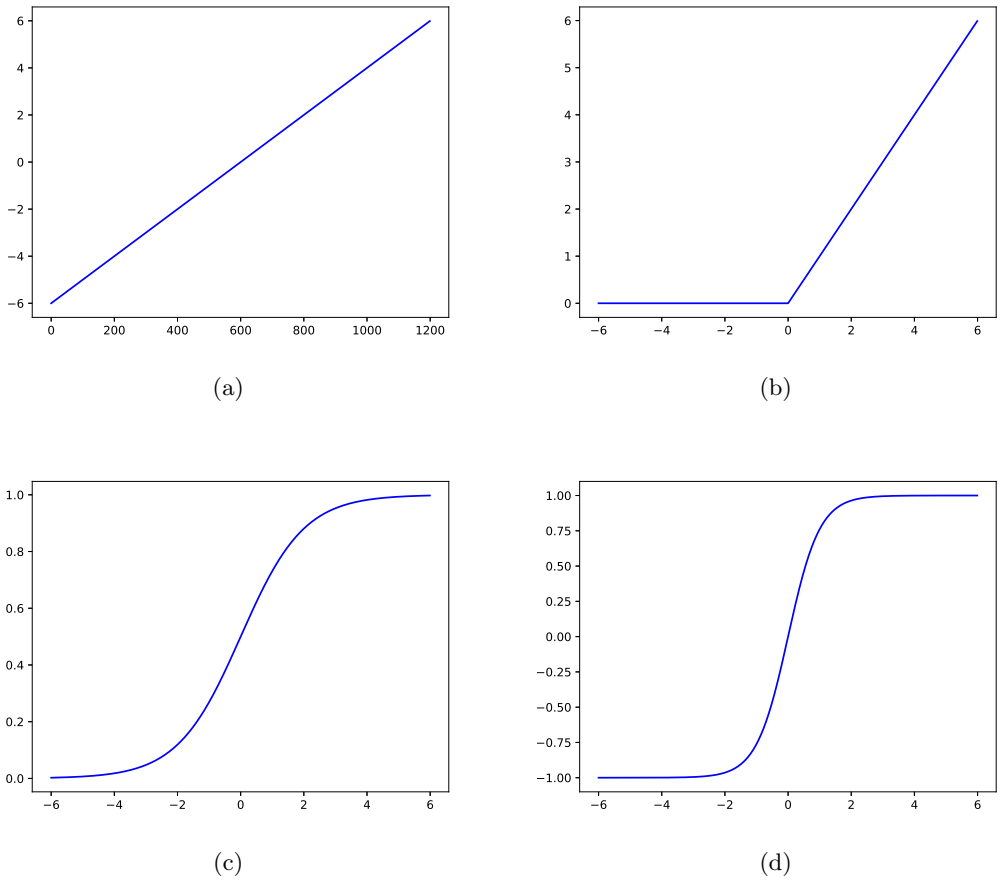


Figure 3.1: (a) Linear, (b) ReLU, (c) sigmoid and (d) tanh activation functions.

RNNs include the weight matrix (U) that is applied to the former output of the RNN h_{k-1}.

The unfolding of a RNN consists of converting the RNN in a multilayer perceptron with as many layers as the length of the sequence input (τ time steps). The multilayer perceptron [41] is the generalization of the perceptron (see Equation 3.1) for the case of multiple layers. Figure 3.3 shows a representation of the unfolding of Figure 3.2 for a $\tau = 2$. By using Equation 3.2 the process is shown in Equation 3.3.

$$h_k = \varphi\left(Wx_k + U\left(\underbrace{\varphi\left(Wx_{k-1} + Uh_{k-2} + b\right)}_{h_{k-1}}\right) + b\right) \tag{3.3}$$

3.2.1 Back-propagation through time

The internal parameters of the ANN are initialized with a random value. Therefore, the objective of training an ANN is to modify its internal parameters (weights and bias) to produce a system's output that approximates closely to the desired value. The back-propagation algorithm [55] is a method to find the local minimum of a function. Therefore, to train the ANN is needed a function that assess the performance of

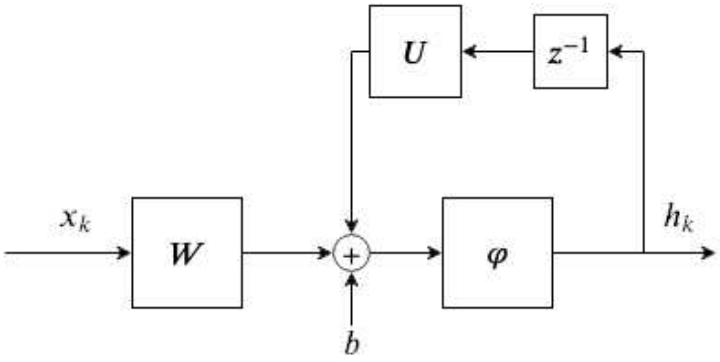


Figure 3.2: Computational graph of a RNN [59].

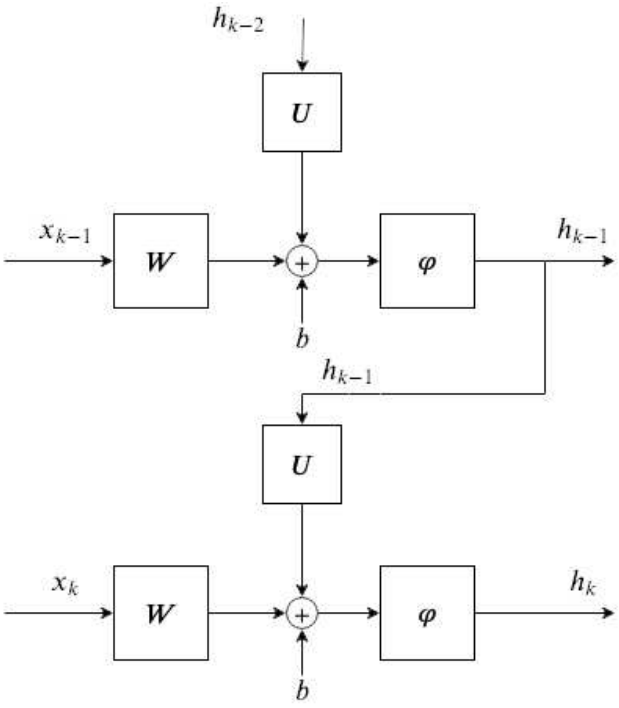


Figure 3.3: Unfolding of a single layer RNN [59].

the model. This performance function is also called loss function. Finding the minimum of this loss function, provides the optimal internal parameters to the ANN and consequently it will perform better in the assigned task. The most common loss function (L) applied in the back-propagation is the mean square error (MSE).

The back-propagation algorithm propagates the error from the loss function (L) by computing the gradient of L. Then, it updates the weights and biases correspondingly to the gradient of L using the stochastic gradient descent (SGD), more sophisticated methods are stated in 4.4.2. In addition to the gradients, it is needed a coefficient called the learning rate (α) to update the parameters, as shown in Equation 3.4. The learning rate (α) is a decisive parameter to train the ANN, its value determines the time it takes to reach the minimum of the function. In addition, a value of α too high may end with the algorithm not being able to converge to find a solution.

$$\begin{cases} W(r+1) = W(r) - \alpha \frac{\partial L}{\partial W(r)} \\ b(r+1) = b(r) - \alpha \frac{\partial L}{\partial b(r)} \end{cases} \tag{3.4}$$

However, RNNs cannot be trained directly with the back-propagation algorithm [55] because the gradients cannot be computed with it. Real-time current learning (RTRL) and back-propagation-through-time (BPTT) are two modified methods of the back-propagation algorithm that are available to train RNN. Both methods provide the same gradients, the main difference is the process followed to calculate them. This Thesis will focus in the BPTT algorithm.

The BPTT algorithm can be seen as applying the back-propagation algorithm to the unfolded RNN [25]. To see how the BPTT behaves, it will be applied to the single layer RNN defined by the Equation 3.2.

The steps are the same as in the back-propagation algorithm [55]. However, now the input vector (x_k) is a sequence from $k = \{1, \ldots, \tau\}$ and the output vector is h_k. The BPTT starts the calculation of the gradient at the last point of the sequence τ, and then it continues working backwards through the whole sequence. The BPTT as the back-propagation algorithm is resolved in three steps: the calculation of the loss function, the application of the chain rule to calculate the gradients and, lastly, the updating of the RNN parameters. The back-propagation algorithm is a iterative process that stops when the minimum of the loss function is founded and the index r denotes the actual iteration, see Equation 3.4.

The first step consists in compute the loss function. The loss function assess the error between the predictions made by the ANN and the correspondent targets (see Section 3.1). In this Thesis, it has been selected the mean square error (MSE), see Equation 3.5.

$$L = \frac{1}{N} \sum_{n=1}^{N} (l(n))^2 \tag{3.5}$$

where $l(n) = [t(n) - h(n)]$, is the vector that measures the error of the predicted output of the RNN ($h(n)$) and the correspondent target ($t(n)$), being n the specific sample [59].

Before continuing, let $A \circ B$ be defined as the element wise product of matrices A and B of same dimension, with elements given by:

$$(A \circ B)_{ij} = (A)_{ij}(B)_{ij} \tag{3.6}$$

Next step consists of back-propagating the error through the network. The chain rule of calculus is used to compute the gradients of the error because the parameters of the RNN are an indirect function of the loss function [16]. Then, the gradients of L are calculated as shown in Eq 3.7.

$$\begin{cases} \frac{\partial L}{\partial W(r)} = \delta(r) x_k(r) \\ \frac{\partial L}{\partial U(r)} = \delta(r) h_{k-1}(r) \\ \frac{\partial L}{\partial b(r)} = \delta(r) \end{cases} \tag{3.7}$$

where $h_{k-1}(r)$ is the output of the time step (k-1) at the iteration r, $k = \{1, \ldots, \tau\}$. $\delta_\tau(r)$ is the sensitivity at the last time step. The sensitivity can be thought of as an intermediate variable that simplifies the calculus and the notation in the back-propagation algorithm, and is computed as shown in Eq 3.8:

$$\begin{cases} \delta_\tau(r) = -2(l_k(r)) \circ \dot{F}_\tau\left(z_\tau(r)\right) \ \ , \ for \ \ k = \tau \\ \delta_k(r) = \left(\dot{F}_k\left(z_k(r)\right) \circ (U(r))^T\right) \delta^{k+1}(r) \ \ , \ for \ \ k = \{\tau - 1, \ldots, 1\} \end{cases} \tag{3.8}$$

where being $z_k(r)$ the output of the RNN in the k time step without the activation function, as shown in Eq 3.9:

$$z_k = W x_k + U h_{k-1} + b \tag{3.9}$$

where $\dot{F}_k(z_k(r))$ is the diagonal matrix shown in the Equation 3.10, being $\dot{\varphi}$ the derivative of the activation function in Eq 3.2.

$$\dot{F}_k = \begin{bmatrix} \dot{\varphi}\left(z_1\right) & \cdots & \cdots & 0 \\ \vdots & \dot{\varphi}\left(z_2\right) & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \dot{\varphi}\left(z_{\delta_k}\right) \end{bmatrix} \tag{3.10}$$

Once the sensitivities are calculated, the last step is to update the parameters of the RNN:

$$\begin{cases} W(r+1) = W(r) - \alpha \left(\sum_{k=1}^{\tau} \delta_k(r) \left(x_k(r)\right)^T\right) \\ U(r+1) = U(r) - \alpha \left(\sum_{k=1}^{\tau} \delta_k(r) (h_{k-1}(r))^T\right) \\ b(r+1) = b(r) - \alpha \left(\sum_{k=1}^{\tau} \delta_k(r)\right) \end{cases} \tag{3.11}$$

It is possible to notice in Equation 3.11 that the gradients for the parameters are summed at each time step, because the parameters are shared across the time steps [25].

Finally, it is important to note the problem of long-term dependencies that were explored by [22] and [8]. When the gradients are propagated over too many time steps, the gradients tend to vanish (very small value) or to become unstable (becoming a very huge value). This problem makes it very difficult to learn long dependencies for the RNNs.

In RNNs, the same activation function is applied during multiples times. It produces a highly non-linear behavior. There are some activation functions that produces this non-linear behavior that boost the vanishing and exploding gradient phenomenon, for example, the sigmoid and the tanh, see Fig 3.1c and Fig 3.1d respectively.

3.2.2 Long Short-Term Memory neural networks

The long short-term memory neural network were introduced by [23]. LSTMs are a special type of RNN developed specifically to solve the vanishing and exploding gradient problem. Thus, being able to learn long-term dependencies as well as short-term dependencies. An LSTM cell (see Fig 3.4) is composed of 4 layers, also called gates, that interact with the cell state (c_k), the output of the LSTM (h_k), also called the hidden state, and between them.

The cell state is the memory of the cell and maintains the information through the unfolding of the LSTM cell. The inputs to the four gates are the correspondent x_k vector, where k is a sequence from $k = 1, \ldots, \tau$, and the previous outputs (hidden states) from the LSTM (h_{k-1}).

The four gates are the forget gate (f_k), the input gate (i_k), the new cell state candidate gate ($\widetilde{c_k}$) and the output gate (o_k). Each gate has its own parameters, biases (b), weight for the input value (W) and weight for the hidden state value (U), and the sub-index used denotes to what gate corresponds. The behaviour of a LSTM cell is the following:

1. The forget layer decides what percentage of the information from the cell state has to be left behind and which should remains in it. The activation function of this gate is a sigmoid (σ) (see Fig 3.1c). Therefore, the output of the forget layer is between 0 and 1. The output of the forget layer (f_k) is multiplied element-wise (see Eq 3.6) to the cell state (c_k). This process is the one that erases the information of the cell state. This process is the same in the input and output gate that also have a sigmoid as activation function.

$$f_k = \sigma\left(W_f x_k + U_f h_{k-1} + b_f\right) \tag{3.12}$$

$$c_{f_k} = f_k \circ c_{k-1} \tag{3.13}$$

2. The next step consists of updating the new information to the cell state, c_{i_k}. The new cell state candidate gate ($\widetilde{c_k}$) proposes the new information that may be added to the memory of the cell (see Equation 3.15). The activation function is a hyperbolic tangent (tanh)(see fig 3.1d), so this gate is, also known as the tanh gate. The input gate decides what percentage of the new information is going to be stored in the memory of the cell. This is done through an element wise multiplication such as in the previous step (see Equation 3.14). The result from these operations is the new information, see Equation 3.16.

$$i_k = \sigma\left(W_i x_k + U_i h_{k-1} + b_i\right) \tag{3.14}$$

$$\widetilde{c_k} = \tanh\left(W_i x_k + U_i h_{k-1} + b_i\right) \tag{3.15}$$

$$c_{i_k} = i_k \circ \widetilde{c_k} \tag{3.16}$$

3. The new cell state is updated with the information provided by the two previous steps, as shown in Equation 3.17.

$$c_k = c_{f_k} + c_{i_k} \tag{3.17}$$

4. The output h_k is obtained through two mechanisms. First, the cell state (c_k) is modified with an hyperbolic tangent (tanh) activation function. To conclude, the output gate selects the percentage of the data that is going to be output by the cell.

$$o_k = \sigma\left(W_o x_k + U_o h_{k-1} + b_o\right) \tag{3.18}$$

$$h_k = o_k \circ \tanh\left(c_k\right) \tag{3.19}$$

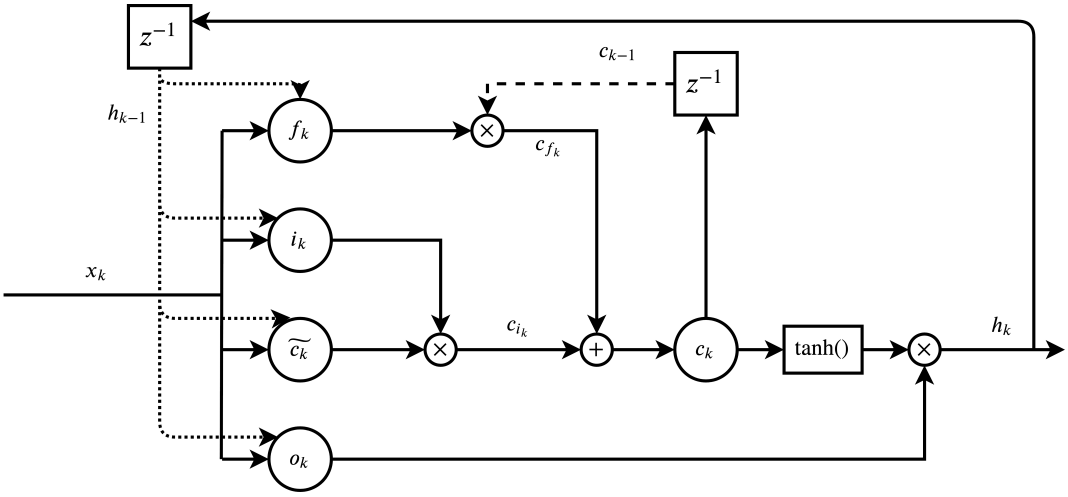


Figure 3.4: Computational graph of a LSTM cell [59].

Chapter 4

Design Process

This chapter describes the design process of this Master's Thesis. First, it presents the development environment used to develop the glucose predictor and the concept of computational graph.

Then, the dataset used in this thesis is presented. Moreover, the pre-processing techniques used and how the data is arranged to train and validate the model proposed is also presented. The neural network architecture proposed for the glucose predictor is shown as well. To conclude, the process of searching the optimal parameters of the neural network is presented.

4.1 Development environment

The chosen framework to implement the predictor is *PyTorch* [44]. *PyTorch* is an open source platform to develop neural network models and it is deeply integrated with python. Two main features from *PyTorch* are worth noting: tensors and dynamic computational graphs.

The *PyTorch* tensors are represented similarly to *NumPy* [43] as n-dimensional arrays. One of the advantages is that PyTorch tensors can be computed on Graphical Processing Units (GPU) with the Compute Unified Device Architecture (CUDA) and this can help to reduce computational time.

First, let's explain what is a computational graph. A according to [59] computational graph represents a mathematical expression and is composed of the mathematical operations, the inputs and the outputs. Mathematical operations are represented by nodes, inputs are the variables that feed the mathematical expression and outputs are the solutions of the mathematical operations [59]. The computational graphs paradigm is implemented in two ways. First, static computational graphs, and second, dynamics computational graphs. In the static computational graph approach, the graph is created before running the program. This is the approach used in frameworks such as TensorFlow. In contrast, *PyTorch* uses the dynamic computational graph paradigm. In *PyTorch*, the computational graph is declared with the Autograd package. Autograd creates the computational graph automatically when model operations are performed. This automatic process is known as a dynamic computational graph. Therefore, the dynamic computational graph can be different at each iteration because it is created on each iteration.

Computational graphs are used by the framework to calculate the back-propagation

algorithm [52] [55]. The Listing 4.1 and Listing 4.2 show how *PyTorch* computes the back-propagation algorithm for a simple neural network example. In this example, it is possible to see that the computational graph is created in the moment of the forward propagation (MLP(x)), the first step of the back-propagation algorithm. Once the gradients are calculated (second step of the back-propagation algorithm) with (loss.backward()), the computational graph is freed. The final step of the algorithm is when the parameters are updated with (optimizer.step()). Also, it would be possible to update the parameters by implementing the expression with tensors. Prior to presenting the example of how the back-propagation algorithm is carried out, following sections will demonstrate the different ways to create ANNs in *PyTorch*.

PyTorch allows ANNs to be developed from scratch using Tensors, but also includes torch.nn class that provides built-in functions and classes to create ANNs with a more comprehensible, concise and scalable approach. It also includes built-in activation functions, such as ReLU, sigmoid and tanh (see Fig 3.1), and loss functions, among others features. To analyze the different approaches, a multilayer perceptron with two layers, 12 inputs and 1 output will be built. The first layer will have a sigmoid activation function and the second layer will have a linear activation function. This neural network will be built from scratch, using the built-in functions and classes from the torch.nn class, and, will conclude by using the sequential class of the torch.nn class. Notice that these approaches are not mutually exclusive.

First, let's present the model done from scratch just using *PyTorch* tensors, see Listing 4.1. It is important to note that the tensors need to activate the parameter requires_grad=True, so the *Pytorch* can calculate the gradients for the back-propagation. In this approach the parameters (weights and biases), the activation function (sigmoid), the forward propagation and the updating of the parameters of the ANN have to be specified.

Listing 4.1: Multilayer perceptron and Back-propagation from scratch

```
import torch


class MLP_scratch:
    def __init__(self):
        self.weights_1 = torch.randn(12, 2, requires_grad=True)
        self.bias_1 = torch.zeros(2, requires_grad=True)
        self.weights_2 = torch.randn(2, 1, requires_grad=True)
        self.bias_2 = torch.zeros(1, requires_grad=True)

    def sigmoid(self, x):
        return 1. / (1. + torch.exp(-x))

    def __call__(self, input):
        layer_1 = input @ self.weights_1 + self.bias_1
        layer_1 = self.sigmoid(layer_1)
        output = layer_1 @ self.weights_2 + self.bias_2
        return output
# Initialization
mlp = MLP_scratch() # defining the ANN
x = torch.randn(12) # input
target = torch.tensor([1.])  # target of the ANN
Loss_funct = nn.MSELoss()  # Defines the loss function
lr = 0.01  # learning rate
# Backpropagation
output = mlp(x)  # Output from the model
loss = Loss_funct(output, target)  # the loss is computed
loss.backward()  # Backpropagate the gradients
# Parameter updating
with torch.no_grad():
    mlp.weights_1 -= mlp.weights_1.grad * lr
    mlp.bias_1 -= mlp.bias_1.grad * lr
    mlp.weights_2 -= mlp.weights_2.grad * lr
    mlp.bias_2 -= mlp.bias_2.grad * lr
    mlp.weights_1.grad.zero_()  # Reset the gradients
    mlp.bias_1.grad.zero_()
    mlp.weights_2.grad.zero_()
    mlp.bias_2.grad.zero_()
```

Another way to deploy an ANN in *Pytorch* is using the nn.Module class, see Listing 4.2. This allows more facilities to create more complex ANNs, whose resulting ANN is more readable. In addition to the functions and classes provided by the nn.Module, *PyTorch* also allows the creation of custom functions and layers. nn.Linear provides a linear layer with the expected number of inputs and outputs, and nn.Sigmoid() provides the sigmoid activation function. In addition, the use of the optimizer simplifies the process of parameter updating.

Listing 4.2: Multilayer perceptron and back-propagation using *PyTorch* classes

```
import torch
import torch.optim as optim
import torch.nn as nn

class MLP_functional(nn.Module):
    def __init__(self):
        super().__init__()
        self.Linear_1 = nn.Linear(12, 2)
        self.sigmoid = nn.Sigmoid()
        self.Linear_2 = nn.Linear(2, 1)

    def forward(self, input):
        layer_1 = self.sigmoid(self.Linear_1(input))
        output = self.Linear_2(layer_1)
        return output
# Initialization
mlp = MLP_functional()  # defining the ANN
x = torch.randn(12) # input
target = torch.tensor([1.]) # target of the ANN
optimizer = optim.SGD(mlp.parameters(), lr=0.01)
Loss_funct = nn.MSELoss()  # Define the loss function
# Backpropagation
optimizer.zero_grad()  # Reset the gradients
output = mlp(x)  # Output from the model
loss = Loss_funct(output, target)  # the loss is computed
loss.backward()  # Backpropagate the gradients
optimizer.step()  # Updates the parameters
```

To conclude, let's present another class that is inside the torch.nn class, the sequential class. The sequential class creates the ANN by adding the modules sequentially. This approach presents a more simple way to write the ANN, see Listing 4.3. The back-propagation algorithm is similar to the Listing 4.2 to calculate.

Listing 4.3: Multilayer perceptron using sequential class

```
import torch
import torch.nn as nn

MLP = nn.Sequential(
    nn.Linear(12, 2),
    nn.Sigmoid(),
    nn.Linear(2, 1))
```

The approach to create ANN with Tensors is very flexible and it allows the creation of any kind of ANN. However, this approach is complex and very prone to error. On the other hand, the use of the sequential class is very readable and easy to use, although the creation of a more complex ANN could be challenging. The nn.Module class allows enough flexibility to create complex ANNs and, at the same time, provides a comprehensible, concise and scalable way to build ANNs. Therefore, it is the selected approach.

Other libraries used in the development of this Master's Thesis are: *Numpy* [43],

Matplotlib [24] and *Pandas* [37]. *Numpy* is used to implement some of the metrics to validate the predictors and also because it was needed as intermediary values between *PyTorch* and the other two libraries. *Matplotlib* is used to display the figures that are in this Master's Thesis. Moreover, *Pandas* is used to access and modify the dataset used in this Master's Thesis. To conclude, *MATLAB* [36] has been used to develop the filter that was applied to the signal, see Section 4.2.1.1.

4.2 Dataset

In order to develop and train our glucose level predictor based on LSTMs, a dataset is needed. In this section, the dataset used and a brief introduction of how the data was acquired is presented. This experiment and the analysis of the data was made after the collection of the data done in [48]. The dataset utilized in this Master's Thesis is the Dataset 2 [48]. This dataset was also used in [49].

The experiment is implemented as a three weeks randomized crossover with the participation of twelve patients, six men and six women. Three variables were acquired from the subjects: glucose levels that were obtained from a CGM sensor (Guardian® Real Time, see Table 4.1) and from a glucometer, insulin bolus from their insulin pump and the food intake. The devices used in the study were the Guardian® Real Time, a glucometer to do the calibrations for the CGM sensor, the insulin pump owned by the subject and a PDA or a paper sheet to record the food intake. The phases of the experiment were: training, experimental, shift and control.

Guardian® Real Time Medtronic-Minimed	
Type of sensor	Minimally invasive
Applied technology	Glucose oxidase enzyme (Enzymatic)
Calibrations required	2 per day
Life time	72 hours on average
Sampling period	5 minutes
Measurement frequency	Continuous

Table 4.1: Main characteristics of Guardian® Real Time [48]

After the experiment, all the data from the twelve patients was gathered and analyzed. The data from the CGM sensor, Guardian® Real Time, was directly downloaded from the sensor so the experimenter could assume the data as reliable. The same happened with the data from the insulin pump. However, the food intake data was filled manually so it needed a study to validate if the data was reliable. That study compared the insulin bolus from the insulin pump with the food intake in order to verify if the data was reliable. According to this study, some of the registers from the food intake were missing, and others were unreliable. For the missing data, the main meals were estimated using the information of the insulin bolus. However, not all the information could be estimated so all the information that was missing and unreliable data were eliminated.

After validating the information, the database had 58 complete profiles from eight

different patients with information about the glucose levels from the CGM sensor, the capillary glucose blood levels from the glucometer (used for the calibration of the CGM sensor), the insulin data from the insulin pumps and the food intake converted into grams of carbohydrates (see a patient example in Figure 4.1). Each profile has the information of one whole day. Each day has 288 samples with a sampling period of 5 minutes, see Table 4.1.

The implementation of the glucose level predictor is the objective of this Master's Thesis. Three parameters from the previous experiment will be used in this thesis: the glucose levels from the CGM sensor, the insulin data from the insulin pump and the the food intake converted into grams of carbohydrates. The glucose values from the glucometer will be used automatically by the Guardian® Real Time to calibrate the output. These three parameters are selected because they are the mains parameters to control the glucose concentration in the body, see Chapter 2. Another important parameter not involved in the experiment, was physical exercise. This data was not recorded. Another issue to note is that all these variables are time dependent. The three parameters provided have the same sampling period, 5 minutes, and are acquired simultaneously. Therefore, the time will not be fed to the predictor because RNNs are able to learn the time dependency by themselves.

The database obtained from the previous experiment has the variability needed to train and evaluate a prediction model [48]. The variability among patients is wide enough to provide different situations to the predictor proposed in this Master's Thesis.

4.2.1 Data preprocessing

The predictor proposed was fed with three parameters: the glucose data from the CGM sensor, the insulin data from the insulin pumps, and the food intake, see Figure 4.2. In this Section, these three parameters and the pre-processing techniques applied to the raw data provided are presented.

Data pre-processing is a different set of techniques that transform raw data into a suitable form needed to increase the performance of ANN. The raw data obtained is often incomplete or it is not suitable to train the ANN. That is why data pre-processing is almost always necessary when working with ANN. Another issue to consider is that the pre-processing techniques have the risk of eliminating the relevant information presented in the data.

4.2.1.1 Glucose preprocessing

There are three main artifacts that can be highlighted in the raw glucose data. There are missing values (values equal to zero, see Figure 4.3), jumps in the signal due to the calibration (see Figure 4.4) and noise in the CGM signal. Therefore, to use the glucose data, the missing data from the sensor needs to be recovered and the impact of the calibrations needs to be reduced, as well as the noise distortions.

First, let us focus on the missing values of the signal. According to [48], this loss data may be caused by a number of factors. One of these factors could be a problem with the connection between the CGM sensor and the receptor. Another factor could be a problem that happens during the calibration. For example, if the

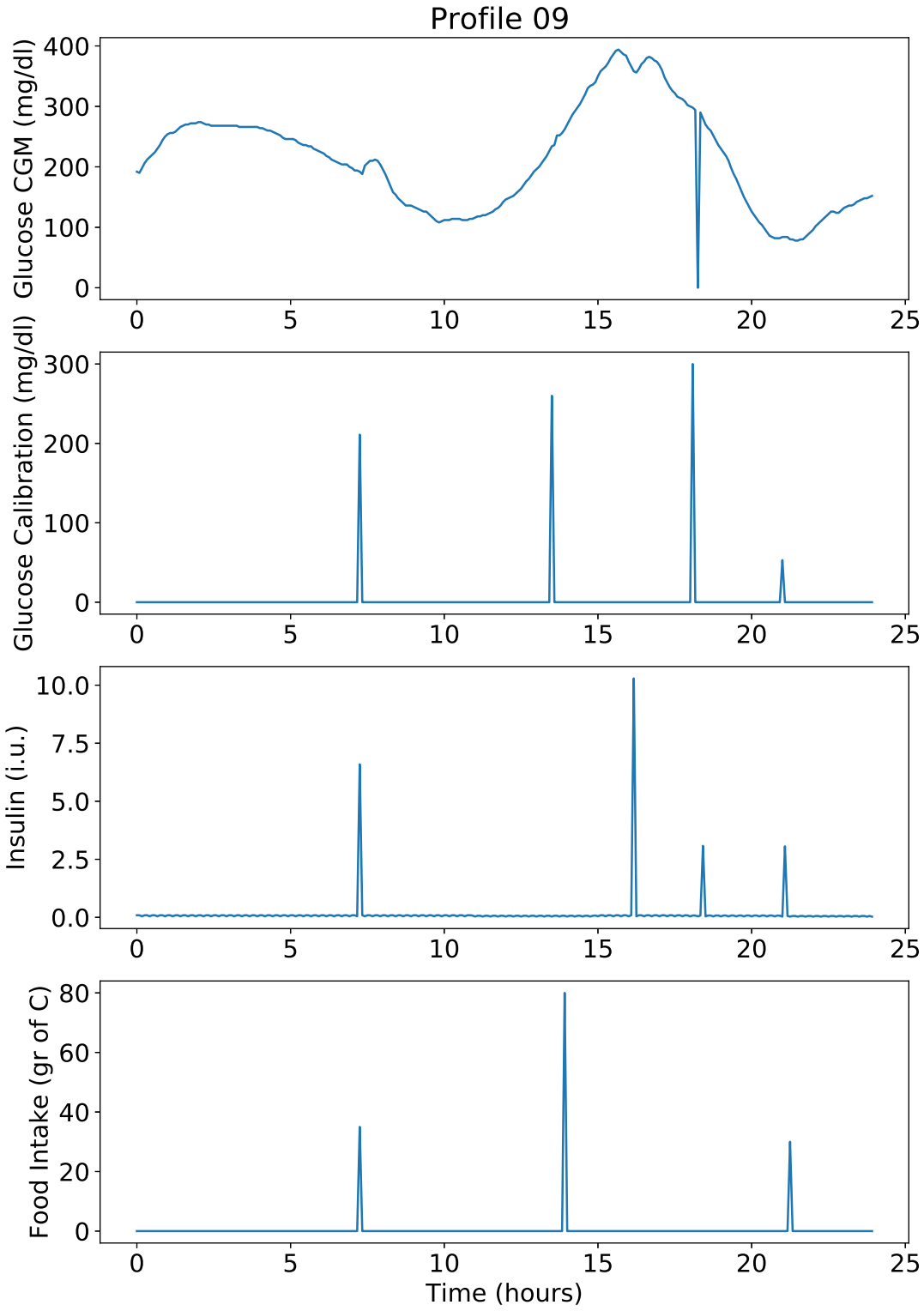


Figure 4.1: Representation of profile 9 of the Patient 2 from the database

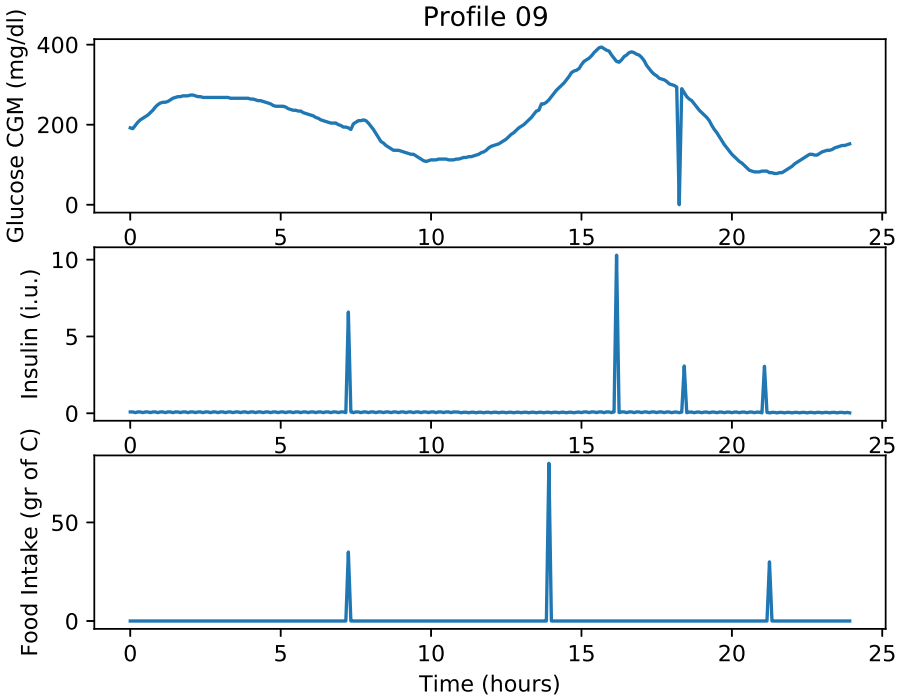


Figure 4.2: Representation of the raw parameters to feed the RNN

sensor measurement is far from capillary blood glucose level then it stops sending measurements until it calibrates itself. The selected method to correct the missing values is the interpolation [48].

Figure 4.3 represents a profile with missing values and its interpolation.

According to [48], the Guardian® Real Time needs at least two calibrations during the day, see Table 4.1. These calibrations are the blood glucose concentration measured by a glucometer. Users are responsible to upload the calibration measurements into the device. Once the calibration is inserted, the internal parameters of the CGM sensor starts to update to the approximate output of the blood glucose concentration measurements from the glucometer. This process may produce some discontinuities in the signal, which impairs the performance of the predictor [48], see Figure 4.4.

To compensate these distortions, a threshold of 30 mg/dl has been defined. Only jumps bigger than or equal to the threshold are compensated. The process is done empirically, and it proceeds as follows: if a distortion that is bigger than or equal to the threshold is found, then it checks if there is an associated calibration value that has produced this distortion. Once a calibration value corresponds with an artifact, then a linear transformation is applied to the signal to correct the distortion. Figure 4.5 shows an example of the results of the process.

When performing this process, the following conclusion has been reached: the 57th profile, that corresponds to the patient 10, has been dropped from the study because it did not have calibration data and the distortions could not be corrected.

The CGM signal is very noisy, so to reduce the noise and smooth the data from CGM signal, a moving average filter is applied. The Equation 4.1 shows the formula of the moving average filter. Let N be the length of the window where the averaging

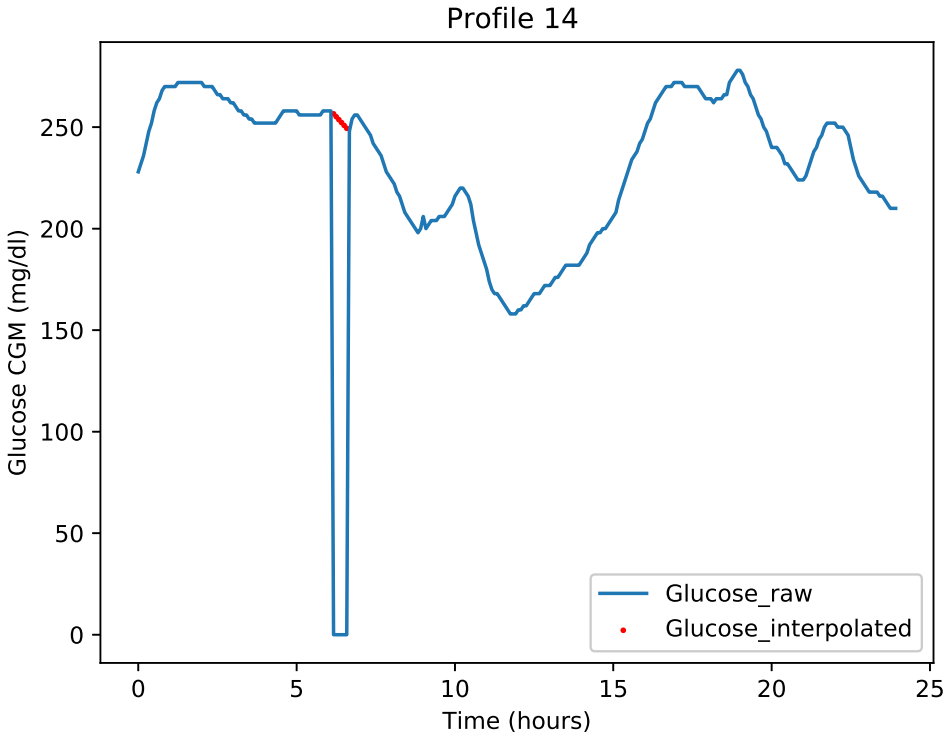


Figure 4.3: Result from the interpolation of Profile 14 from Patient 3

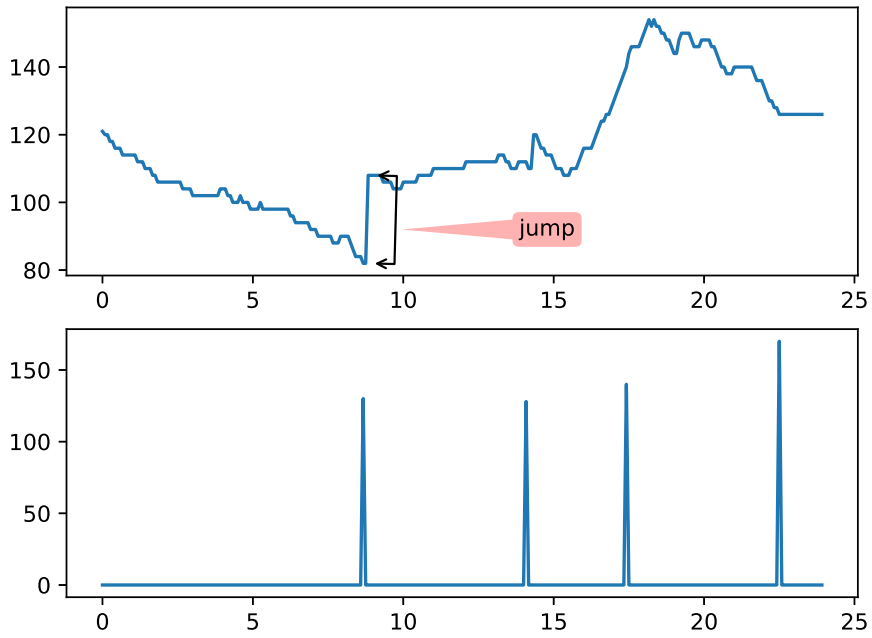


Figure 4.4: Jump produce by the calibration

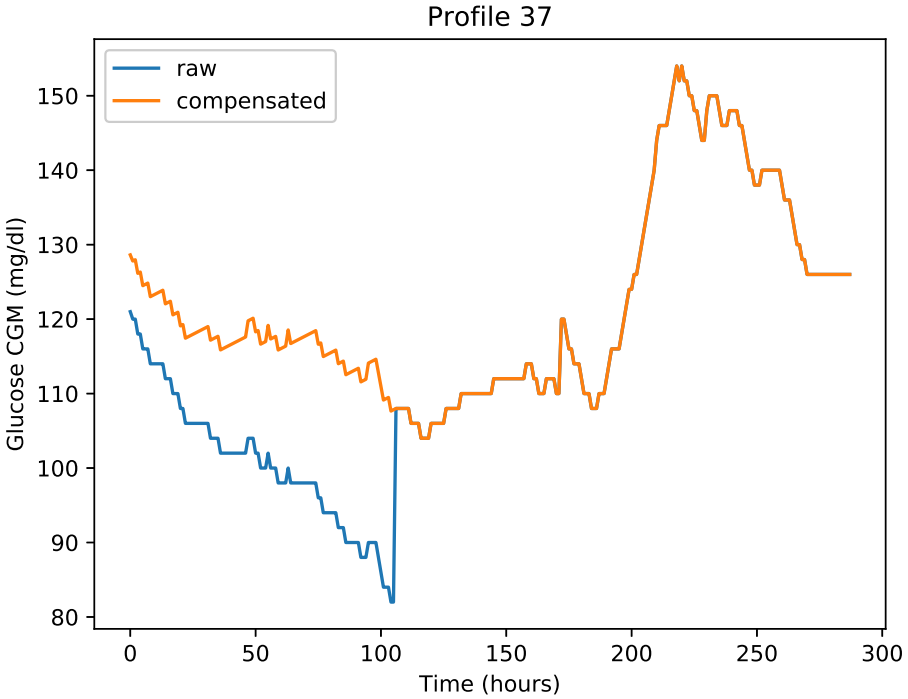


Figure 4.5: Results after correction of the discontinuities caused by the calibration

is applied, x the raw signal, and $\hat{x}$ the filtered signal. The moving average filter is a finite impulse response (FIR) filter, so it only depends on the input parameters and it is a time dependent filter. There are two things to keep in mind when a filter is applied. First, the signal is not modified excessively, so the relevant information is still presented in the signal. Second, the filter provokes a delay in the signal applied. Nevertheless, the benefits of applying a filter outweigh the disadvantages of using it.

Taking all this into account, the chosen moving average filter has a length $N = 5$. This allows the signal to be cleaned and avoids deleting relevant information (see Figure 4.6).

$$\hat{x}\,(i) = \frac{1}{N}\sum_{n=0}^{N-1} x(i-n) \tag{4.1}$$

4.2.1.2 Insulin preprocessing

Insulin pumps have two kinds of insulin injections: basal insulin and bolus insulin (see Section 2.4.1). The bolus insulin for the meals is several orders of magnitude bigger than the basal insulin. The insulin bolus are represented as an impulse function or Dirac delta function (see Figure 4.7). This may produce the RNN to ignore the value of the bolus of insulin or another unforeseen behaviour. Therefore, a pre-processing technique is needed.

The chosen solution is to transform the bolus of insulin, that is an impulse function, into a rectangular function with the same area. The insulin bolus is

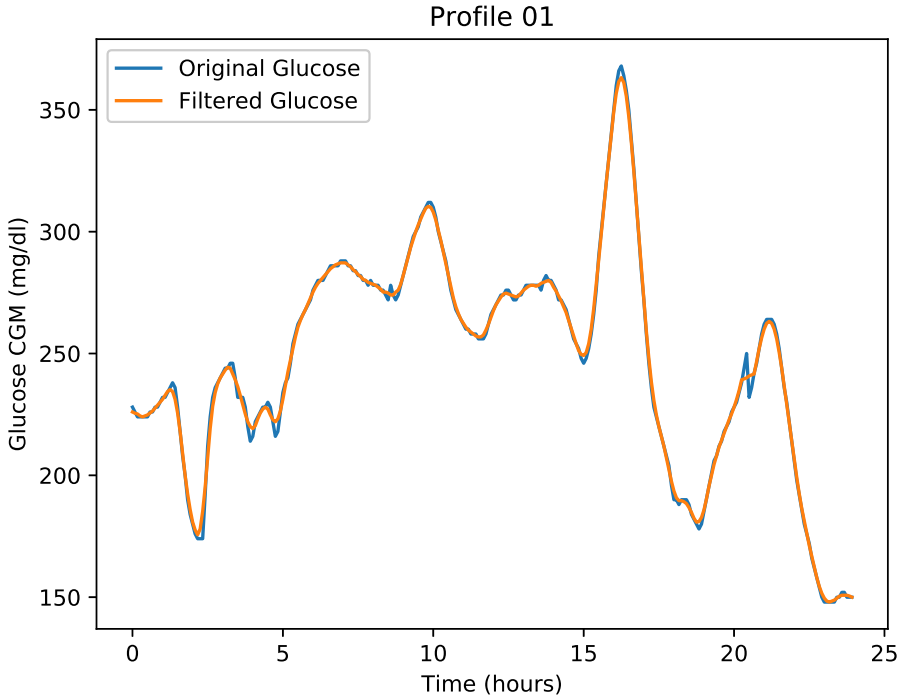


Figure 4.6: Comparison between the raw CGM and the filtered CGM signal

distributed during 12 samples that correspond to 1 hour, because rapid acting insulin used in insulin pumps, tends to acts during 1 hour on average. In addition, the basal insulin is not transformed and it behaves as before. Figure 4.8 shows the results of the pre-processing technique applied.

4.2.1.3 Food intake preprocessing

The food intake was manually provided by the patient. These values were transformed to grams of carbohydrates, representing the amount of carbohydrates eaten by the patient in the meal. Each registered meal is represented with only one value. The representation of this parameter shows that the food intake is also represented as a impulse function or Dirac delta function, see Figure 4.9. The problem is similar to the one stated in the Section 4.2.1.2. Therefore, the solution is also similar.

The food intake parameter is transformed into a rectangular function with the same area that the impulse function had with 12 samples equivalent to one hour. The value of one hour is being established as the average absorption time of glucose for the different types of carbohydrates. Figure 4.10 shows the results of the pre-processing technique applied.

4.2.1.4 Normalization

The predictor proposed in the Thesis is based on LSTMs, which are a type of RNN that have sigmoid and tanh activation functions in their gates (see Section 3.2.2). The value of the sigmoid output is in the range between 0 and 1 (see Figure 3.1c). Therefore, the output of the sigmoid saturate in 1 and into 0. Something similar

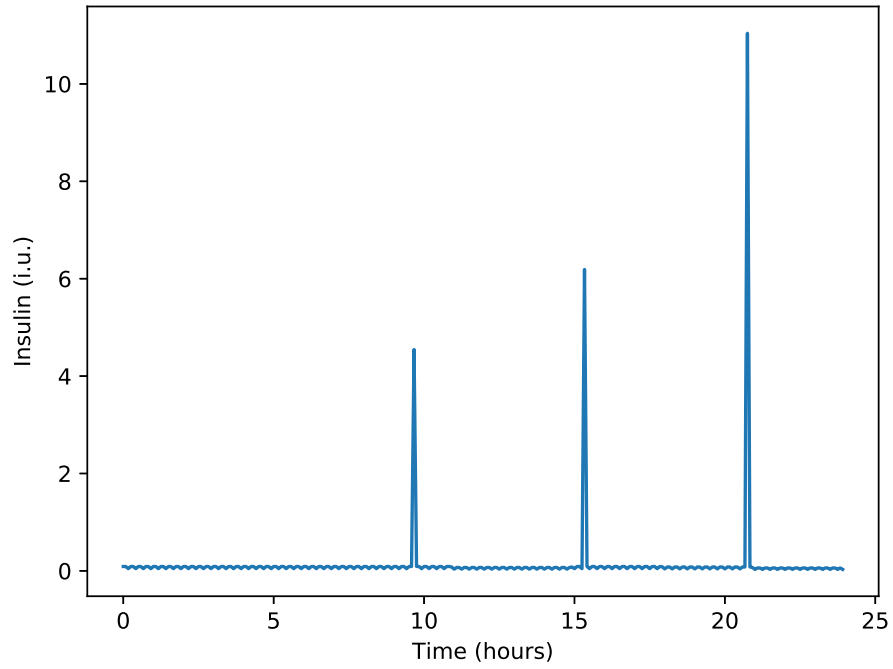


Figure 4.7: Raw insulin input

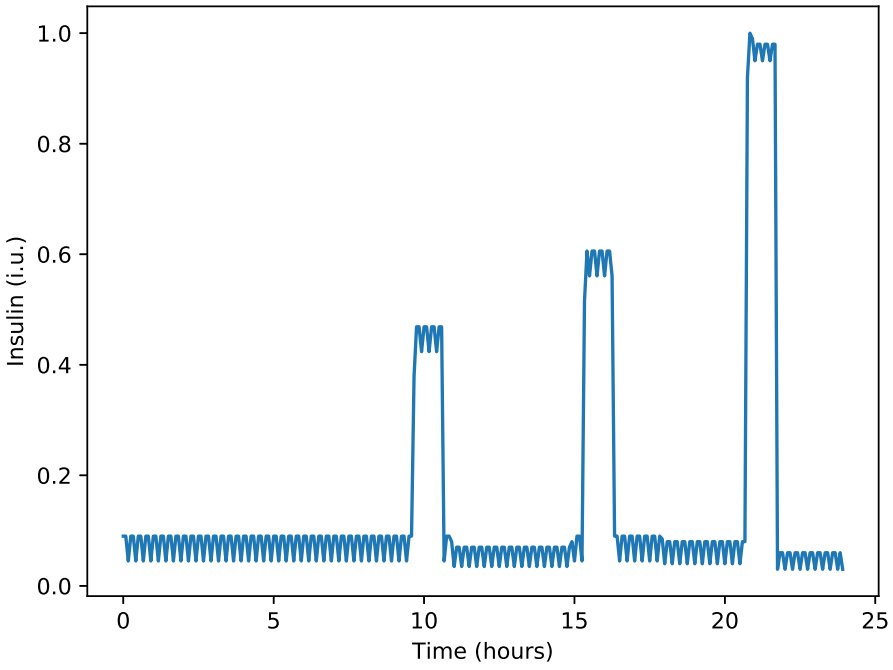


Figure 4.8: Insulin input after preprocessing

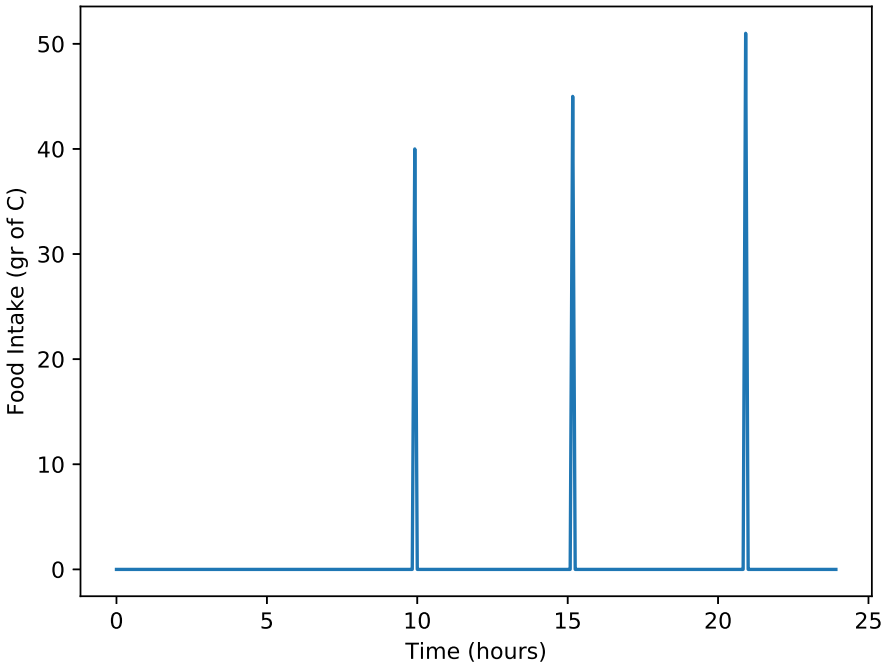


Figure 4.9: Raw food intake

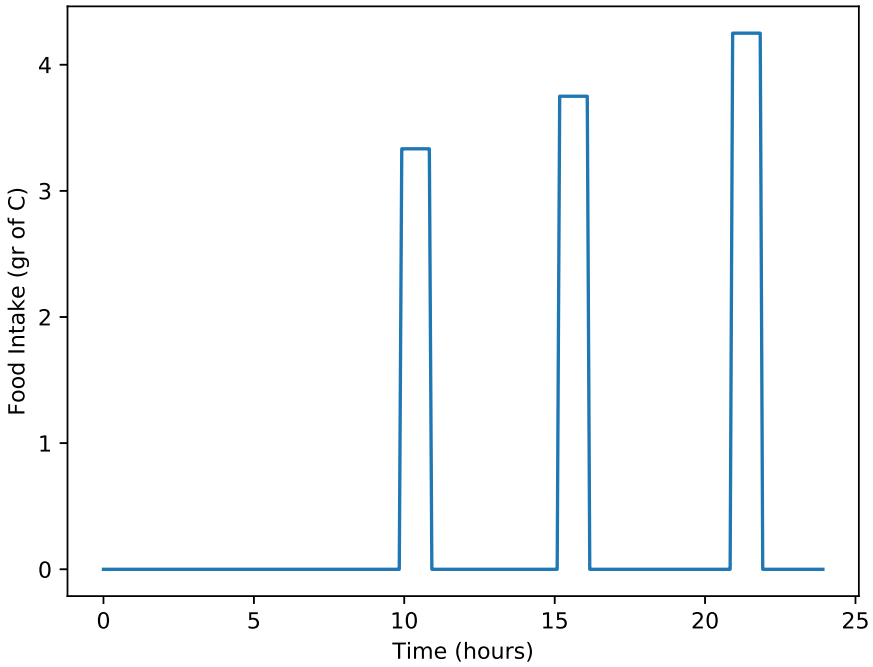


Figure 4.10: Food intake input after preprocessing

happens with the tanh function, but in this case their range is between -1 and 1 (see Figure 3.1d). It is important to note that the range from the sigmoid $(0, 1)$ is more restrictive than the range of the tanh. Therefore, the input parameters to the RNN: glucose from the CGM sensor, the insulin and the food intake, should be in the range of 0 to 1 to prevent unforeseen results and to facilitate the proper learning of the neural network. Moreover, this process is also done to the targets of the RNN model. Sections 3.1 and 4.2.2 explain the targets.

There are multiple ways to normalize the data in order to train an ANN, some examples are: logarithmic normalization, standard normalization or min-max feature scaling. The chosen method has been the min-max feature scaling, because it guarantees the normalized values will be in the range of $(0, 1)$. This method proceeds as follow:

1. For each parameter of the network, it has to be found the minimum value in the dataset for that specific parameter, x_{min}.

2. Similarly, the procedure is the same for the maximum value in the dataset for that parameter, x_{max}.

3. Once that the previous steps are complete, the normalization is applied:

$$\bar{x_i} = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{4.2}$$

   Let x_i correspond to the value of the specific parameter, that is being normalized. Where $i \in \{1, ..., N\}$ being N the numbers of samples in the dataset.

4.2.2 Data arrangement

After finalizing the pre-processing process described in Section 4.2.1 the parameters are prepared to be fed into the RNN. In this section, the process of preparing the data to train and evaluate the predictor based in LSTMs will be presented.

The paradigm of learning used, as previously stated, is supervised learning (see Section 3.1). Then, the data is to be arranged in inputs, $I = \{i_1, \ldots, i_n\}$, and targets, $T = \{t_1, \ldots, t_n\}$. This dataset consists of a time-series of 288 samples distributed every 5 minutes that has already been pre-processed. These time-series are: the glucose (X), the insulin (C) and the food intake (Z). These time series have to be converted in a supervised learning problem, this may be done in a variety of methods. However, for this Thesis, the sliding window or windowing ([29] and [31]) has been selected and it has been applied as follows:

1. Prior to the explanation of the windowing method, it is important to define some important terminology. The input dimension, also known as the sequence of values, determines how many past values of glucose, insulin or food intake are fed as an input to the predictor. The prediction time (PT) states the time at which the prediction will be made. PT states the time of the prediction in minutes, but notice that the data is sampled every 5 minutes, so the prediction time will be a multiple of 5.

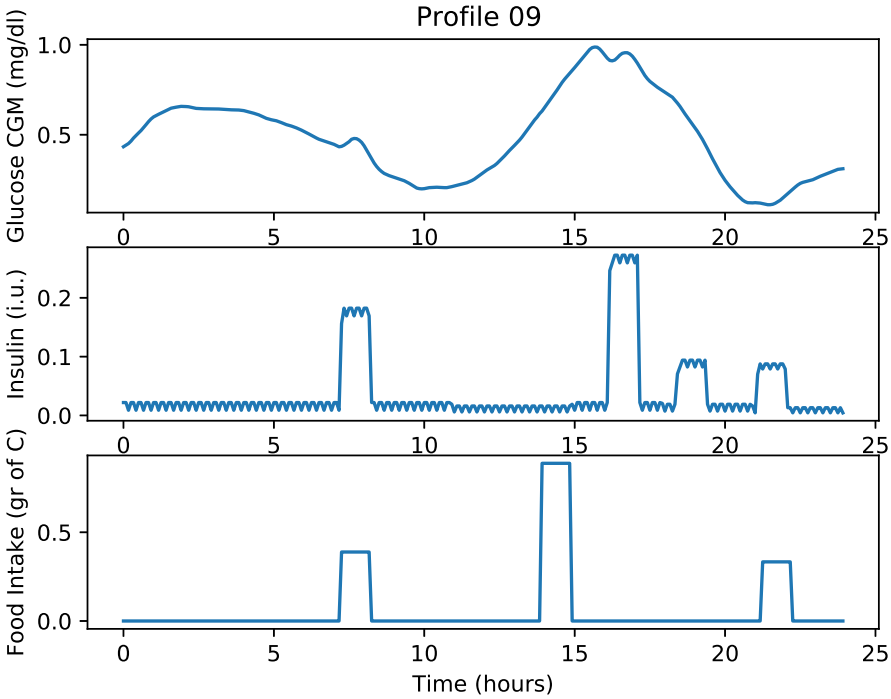


Figure 4.11: Parameters after normalization

2. Additionally, it is important to understand the functions of the glucose parameter. The past values of glucose are an input to the predictor and, also, the future values of glucose are the target of the predictor. Therefore, the time series has to be re-arranged in input values and in target values. The insulin and the food intake parameters are only inputs to the system, so they only have to be re-arranged as input values.

3. To illustrate the process, let the input dimension be 5 and the PT be 10 minutes, which is equivalent to 2 samples, the time series of glucose $X = \{x_1, x_2, \ldots, x_n\}$, the insulin, $w = \{w_1, w_2, \ldots, w_n\}$, and the food intake, $Z = \{z_1, z_2, \ldots, z_n\}$. Then, the data is arranged as illustrated in Figures 4.12 and 4.13, where I are the inputs (see Figure 4.12) and T are the corresponding targets (see Figure 4.13).

   Note that I is a three dimension tensor, where the first one is the new size of the dataset after the arrangement with the sliding window method. The second dimension is the input dimension, that is the number of past values that will feed the predictor. Finally, the third one is the number of input parameters, in this case, three. T is a single dimension tensor with correspondent future glucose value for each row of the Inputs (see Figure 4.12). The inputs may be also represented as in Eq 4.3.

$$I = \{\mathfrak{I}_1, \mathfrak{I}_2, \ldots, \mathfrak{I}_i, \ldots \mathfrak{I}_n\} \tag{4.3}$$

   where each $\mathfrak{I}_i$ is a tensor, that represents an input value to the predictor. Being

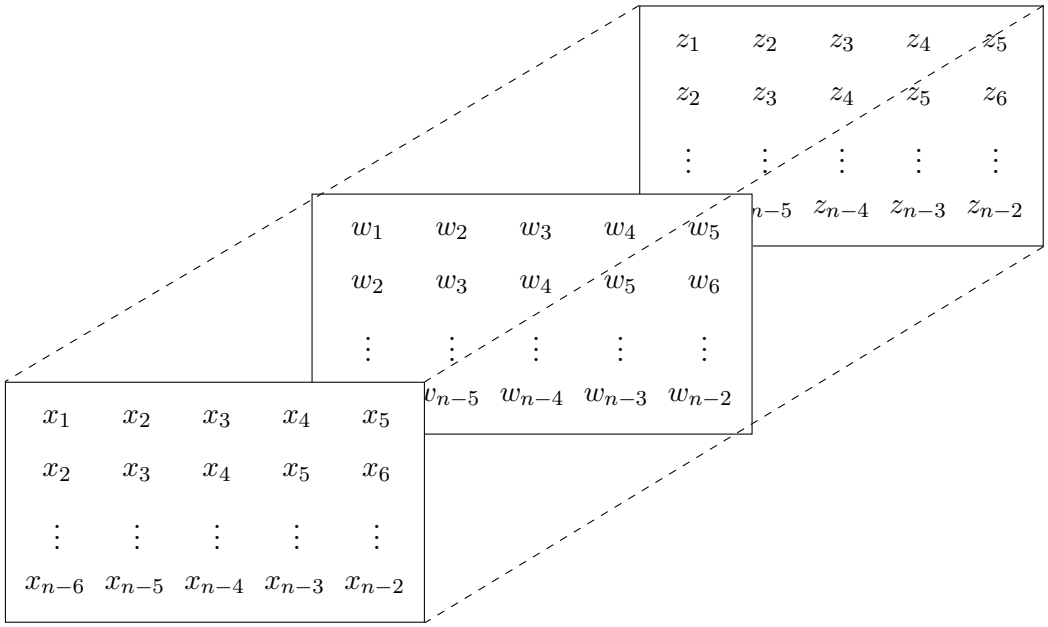


Figure 4.12: Three dimension input tensor that feed the predictor.

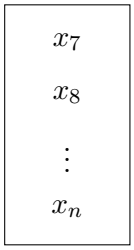


Figure 4.13: Correspondent targets.

$\mathfrak{I}_i$ a sequence of values from $k = \{1, ..., \tau\}$, being τ equal to the input dimension, in this example, 5 time steps, as shown in Eq 4.4.

$$\mathfrak{I}_i = \begin{bmatrix} x_i & x_{i+1} & x_{i+2} & x_{i+3} & x_{i+4} \\ w_i & w_{i+1} & w_{i+2} & w_{i+3} & w_{i+4} \\ z_i & z_{i+1} & z_{i+2} & z_{i+3} & z_{i+4} \end{bmatrix} \tag{4.4}$$

To simplify, let $\mathfrak{i}_k$ be the input vector for the time step k (see Equation 4.5). Then, $\mathfrak{I}_k$ is represented as shown in Eq 4.6

$$\mathfrak{i}_k = \begin{bmatrix} \mathfrak{i}k \\ w_k \\ z_k \end{bmatrix} \tag{4.5}$$

$$\mathfrak{I}_k = \begin{bmatrix} \mathfrak{i}_k & \dots & \mathfrak{i}_\tau \end{bmatrix} \tag{4.6}$$

Once the data in the dataset is converted in a supervised learning problem, the next step is to arrange the dataset into three smaller subsets: training, validation, and the test datasets. It is important to note that the data in each subset only belongs

to that subset, and does not appear in the others. The training dataset holds the data that is used to train the ANN. The ANN makes predictions with the inputs that it receives. Then it compares this prediction with the targets using the defined loss function to calculate the error. Finally, back-propagates the error through the ANN and updates the parameters (see Equation 3.4). The training dataset is the only one where the back-propagation algorithm is applied. Therefore, the training dataset has a great importance because the ANN only learns from this data, thus the training dataset hold the majority of the data.

The validation dataset is also used during the training process to test the performance of the model for unseen data. In this way it prevents overfitting. Overfitting is the overspecialization of the ANN for the data presented in the training dataset, which causes poor performance when different data is presented. The process followed in this thesis is to stop the training when the performance for the validation dataset begins to worsen, this method is called early stopping [46] (see Figure 4.14). The validation data only uses a small portion of the whole dataset.

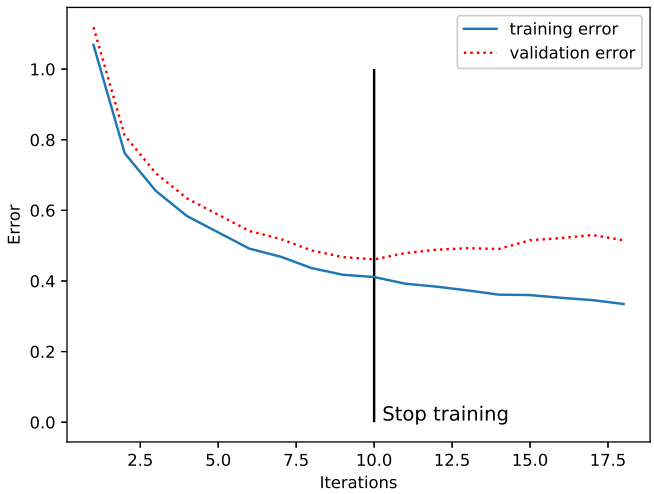


Figure 4.14: Example of the early stopping method.

The purpose of the Test dataset is to evaluate the performance of the fully trained ANN for unseen data. The targets in the test dataset, as well as in the validation dataset, are only used to compare the predictions made by the ANN with the expected targets. There are different metrics used to evaluate the performance of the ANN in the test dataset (see Section 5.1). The amount of data in the test subset should be long enough to ensure that the ANN could be evaluated properly, but leaving enough data for the training dataset.

At the moment the data is distributed in each subset, it is important to note that the data belonging to each patient should be only in one of the datasets, because this data is correlated among them and this could bring misleading results as the data would be biased. Therefore, to ensure the independence of the findings done in this Thesis, the data would be arranged according to that.

Another important issue to consider is that the distribution of the data should be balanced between the number of hard and easy cases to predict. This means that each set (Training, Validation and Test) should have a similar proportional relation between the difficulty of the cases to predict. If this is not addressed, the different subsets will not be representative of the dataset and will not fully comply with their function. In addition to the balance, the different subsets should be as representative as possible of the phenomenon to predict, in this case, glucose concentration. Therefore, each subset has cases with different hypoglycemic and hyperglycemic events, different quantities and times of food intake and insulin injections, and different glucose concentrations, tendency and speed of change in each period of time.

Taking all of this into consideration, the different subsets are distributed as follows: training dataset has 36 profiles from patients 1, 2, 3, 5 and 7, validation dataset has 6 profiles from patient 10 and test dataset has 15 profiles from patients 6 and 9.

4.3 Neural network architecture

In this Section the architecture of the proposed predictor will be presented. The proposed predictor is based in LSTM neural networks, see Section 3.2.2. Here a generic version of the model is presented, and in Section 4.4 the parameters of the model are tuned for each prediction time (PT) proposed. Four forecasts are done, one at each proposed PT: 5 minutes, 15 minutes, 30 minutes and 45 minutes. Notice that the PT at 5 minutes does not have much value for a patient due to its very short notice, so the patient cannot take actions to prevent an hyperglycemia or hypoglycemia. Nevertheless, it is helpful to the designer to give an insight of the best performance that can be achieved.

Furthermore, in this study three values of input dimensions are set: 5 samples, 20 samples and 50 samples. The input dimension is related to the unfolding, as explained in Section 3.2. The input dimension states the number of times that the RNN is unfolded. Therefore, our input tensor ($\mathfrak{I}_i$) will be a sequence of values from $k = \{1, ..., \tau\}$, assuming τ equal to the stated input dimension. This input dimension states how many past samples are taken into account by the model to made the prediction so, 5 samples are equivalent to 20 minutes in the past, 20 samples are equivalent to 1 hour and 30 minutes in the past and, to conclude, 50 samples are equivalent to 4 hours and 10 minutes in the past.

The predictor proposed in the Figure 4.15, is composed of three layers, two LSTM layers and one final linear layer. The LSTM layers (see Section 3.2.2) has 50 neurons each and the linear layer is a perceptron with only 1 neuron (see Equation 3.1) and a linear activation function (see Figure 3.1a). The optimizer algorithm to update the parameters is the Adam [30] and a chosen learning rate of 0.001. These parameters, are also called hyper-parameters and will be optimized in Section 4.4. Only the linear layer will remain as stated in this Section.

Figure 4.15 also presents the unfolding and the data flow of the model. First, the input sequence ($\mathfrak{I}_i$) comes in the first LSTM layer. In this layer the unfolding is processed and the output (h_k^1) is the input to the following LSTM layer. Note that the superscript denotes the layer of the model. The second LSTM layer processes

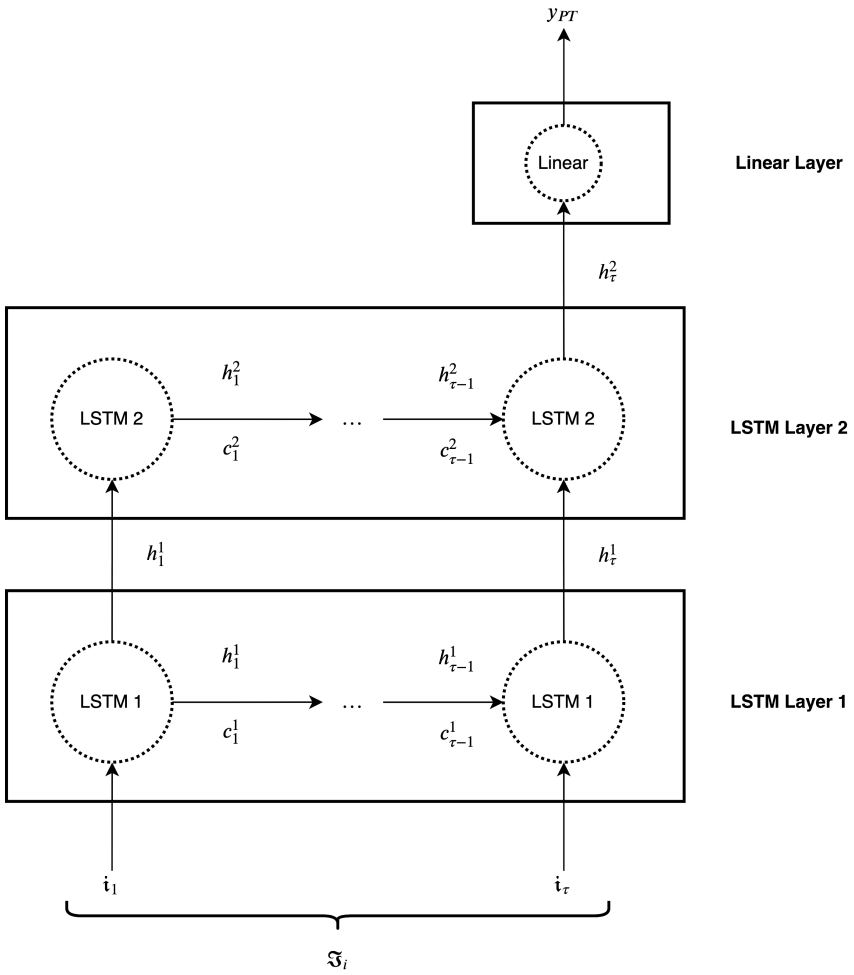


Figure 4.15: Generic architecture version of the predictor proposed.

the information, and provides its output for the last value of the sequence, h_τ^2 to the Linear layer. Finally, the linear layer computes the h_τ^2 to produce the expected value of glucose concentration at the desired PT, y_{PT}.

4.4 Parameter optimization

This section is devoted to find the optimal parameters for the proposed predictor. There are different approaches that may be followed to find the optimal parameters of the neural network. For instance, some of these approaches are: experimental search, Bayesian optimization [61] or using another ANN that performs the parameter optimization, which is another field of the machine learning called AutoML. An example of this is presented in [18].

In this Master's Thesis the chosen method is the experimental search, leaving the others for future research. The proposed method is as follows. For each of the input dimensions selected for the study (5, 20 and 50) and for each of the PT selected (5, 15, 30 and 45 minutes), a number of combinations of the different parameters are stated and then it searches the parameters that produces the best performance. These are considered the optimal parameters for that input dimension and PT. In Section 5.2,

the models with these parameters are trained and assessed, and the results of their performance are shown.

Two metrics are selected to compare the performance of the different parameters. The first one is the error distribution e shown in Equation 4.7 [59]. The other metric used is the root mean square error (RMSE) (Equation 4.8).

$$e = (|t_1 - y_1|, \ldots, |t_n - y_n|) \tag{4.7}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(t_i - y_i)^2} \tag{4.8}$$

where t_i is the expected output (target) and y_i is the prediction made by the model.

To help illustrate the differences between the parameter box-plots that are used to display the error distribution e and bar charts are used to represented the RMSE for each combination of parameters.

The search of the optimal parameters is divided in two parts, the architecture parameters and the parameters related to the training of the proposed predictor. First, the purpose of the architecture parameters search is to find the optimal number of neurons and layers for the predictor. In the search of the training parameters, it will find the best optimizer to update the parameters in the back-propagation algorithm and the optimal learning rate (α) that improves the performance of the model.

4.4.1 Architecture parameters

Once the main architecture is defined and the types of ANN are used, the architecture parameters that define the structure of the proposed model are the number of neurons and the numbers of layers.

In this study, the optimal parameters for the architecture presented in Section 4.3 will be searched. The linear layer will not be modified, its composition remains in 1 layer with 1 neuron, so during this Section when reference is made to the number of layers or neurons, it refers to the LSTM layers of the model.

Each LSTM layer will have the same number of neurons. The number of neurons that are considered in this study are: 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100. While the numbers of LSTM layers are 1, 2 and 3. This gives a total of 30 combinations that are searched to find the best performance for the prediction time (PT) stated (5 min, 15 min, 30 min and 45 min). In addition, a learning rate of 0.001 and the Adam optimizer [30] have been chosen. This processes is repeated for the three different input dimensions chosen 5, 20 and 50.

The distribution error e is shown in Figures 4.16, 4.17 and 4.18 for the input dimension of 5, 20 and 50, respectively. In addition, in Figure 4.19 shows the RMSE for each PT and each input dimension. Figures 4.16, 4.17 and 4.18 show a very similar distribution error e for each model. In addition, Figure 4.19 also shows a very similar performance for each model, however, there some cases where the error is higher. Taken into account the results from both metrics, the best parameters are presented in Table 4.2 with all the optimal parameters.

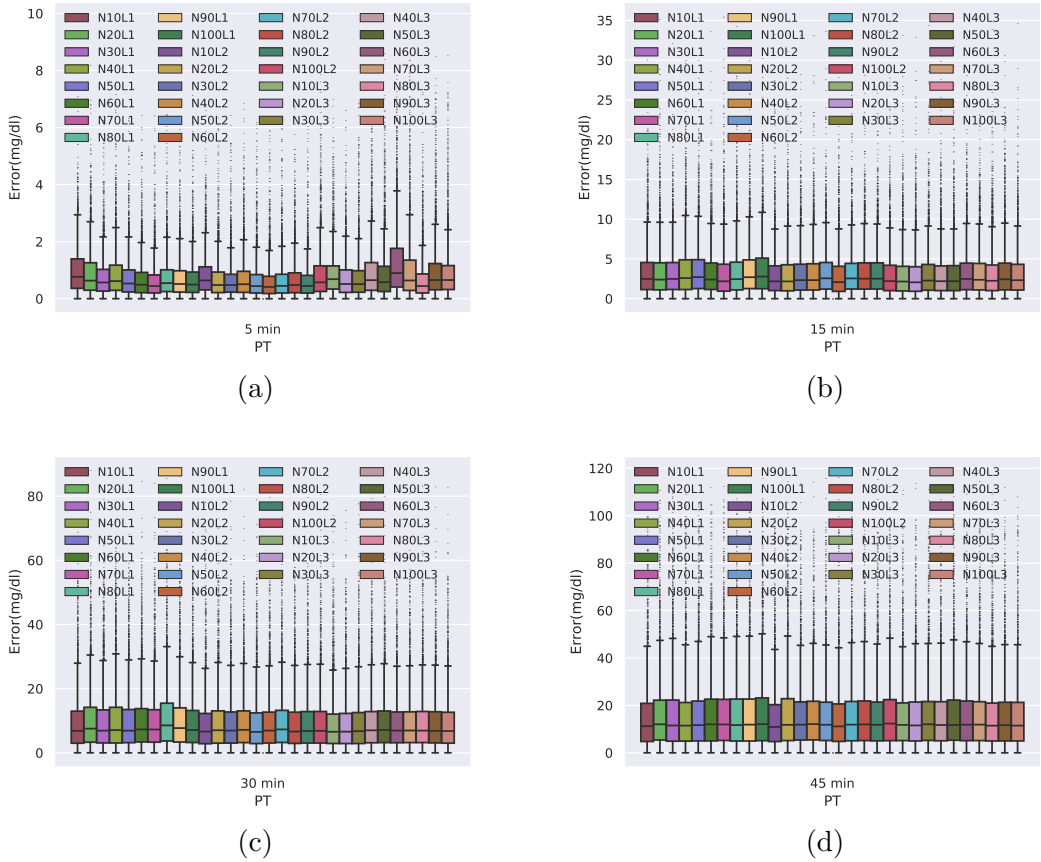


Figure 4.16: Box-plots for the e for the different architecture parameters for a input dimension of 5, being PT (a) 5 min (b) 15 min (c) 30 min (d) 45 min.

4.4.2 Training parameters

Once that the architecture parameters are selected, the next step is to find the optimal combination of parameters involved in the training. The parameters that are included in this search are the optimizer and the learning rate (α) used in the back-propagation algorithm. The selection of these parameters is crucial to achieve the finest training. The more accurate the training of the ANN is, then its performance will be increased in the evaluation when new data is presented.

Two optimizers are chosen to train the proposed predictor: RMSprop [20] and Adam [30]. These optimizers are used to update the weights and biases of the ANN in the back-propagation algorithm. They are an alternative to the SGD explained in Chapter 3.2.1. Therefore, both RMSprop and Adam algorithm use a variation of Equation 3.4. On the other hand, five different learning rates (α) are considered $\alpha = [0.0001, 0.0005, 0.001, 0.005, 0.01]$ to search the best parameters for the training of the ANN. This makes the total of 10 combinations to calculate.

[20] proposes the RMSprop algorithm. This algorithm presents a modification of the SGD that adapt the learning rate depending on the moving average of the squared gradients, $\mathfrak{g}(r)$. Where $\mathfrak{g}(r)$ and the updating of the weights and biases is calculated as shown in Equation 4.9.

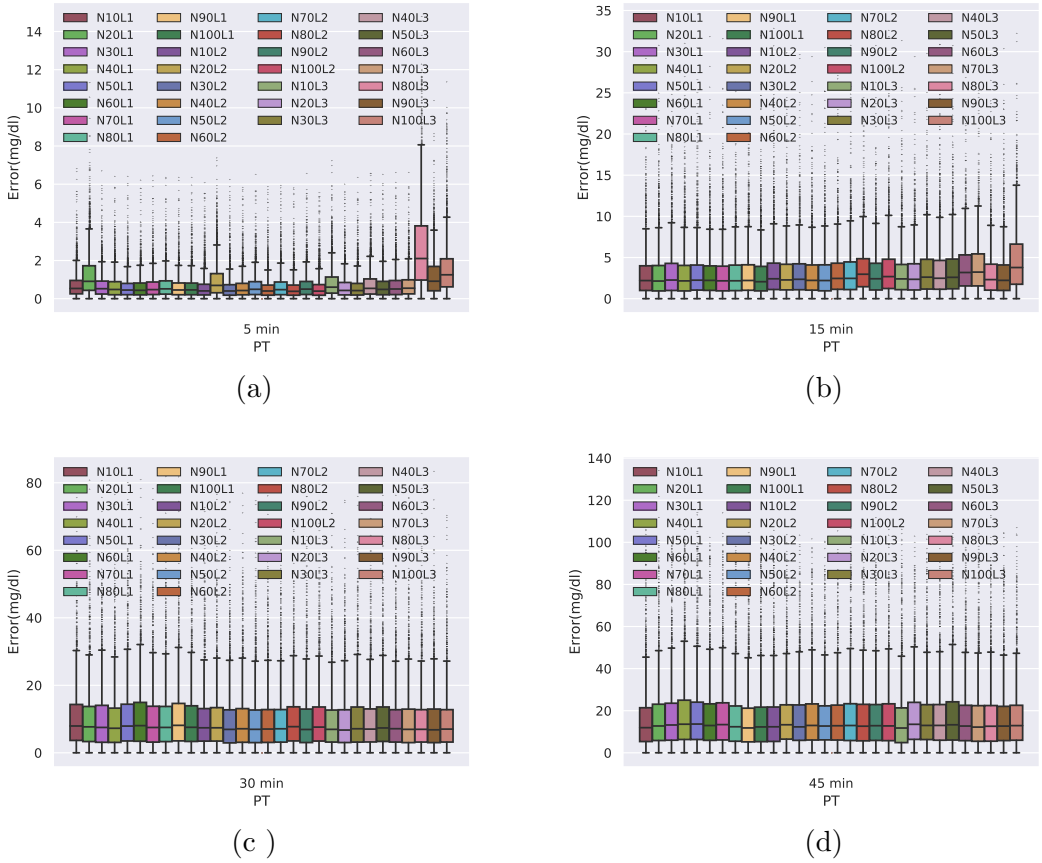


Figure 4.17: Box-plots for the e for the different architecture parameters for a input dimension of 20, being PT (a) 5 min (b) 15 min (c) 30 min (d) 45 min.

$$\begin{cases} \mathfrak{g}_w(r) = \beta \mathfrak{g}_w(r-1) + (1-\beta)\left(\frac{\partial L}{\partial W^m(r)}\right)^2 \\ \mathfrak{g}_b(r) = \beta \mathfrak{g}_b(r-1) + (1-\beta)\left(\frac{\partial L}{\partial b^m(r)}\right)^2 \\ W^{m+1}(r+1) = W^m(r) - \frac{\alpha}{\sqrt{\mathfrak{g}_w(r)+\epsilon}} \frac{\partial L}{\partial W^m(r)} \\ b^{m+1}(r+1) = b^m(r) - \frac{\alpha}{\sqrt{\mathfrak{g}_b(r)+\epsilon}} \frac{\partial L}{\partial b^m(r)} \end{cases} \tag{4.9}$$

where the subscript b and w indicates the gradients for the biases and weights in $\mathfrak{g}$. β is constant, the value chosen is 0.99, and ϵ is another constant that prevents the division by zero, with a chosen value of 10^{-8}. Finally, the index r denotes the iteration.

The Adam algorithm, proposed by [30], in contrast to the RMSprop algorithm uses two moving averages. First, it keeps the moving average of the gradient ($m(r)$), as a biased estimator of the mean of the gradient. Similarly to the RMSprop algorithm, it calculates the moving average of the squared of the gradient ($v(r)$) and it is a biased estimator of the variance of the gradient. Their formulas are shown in Equation 4.10. Note that the subscript b and w indicates the gradients for the biases and the weights and that β_1 and β_2 are two constants defined by the user.

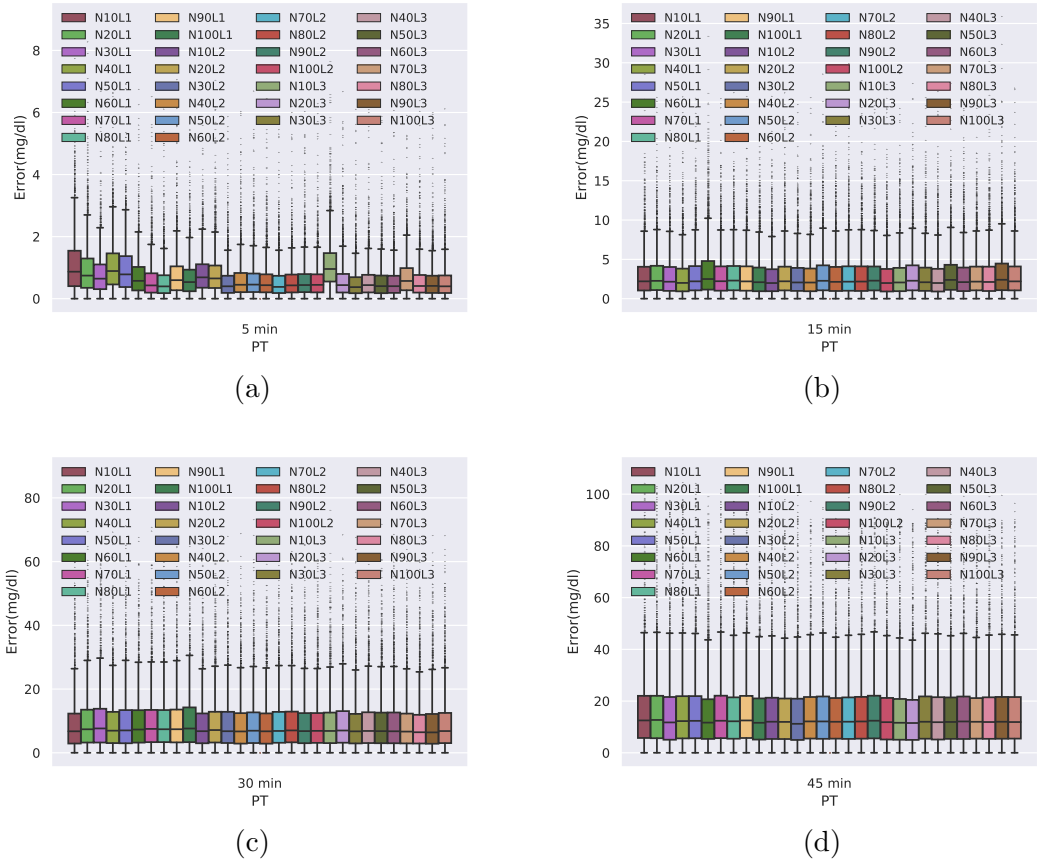


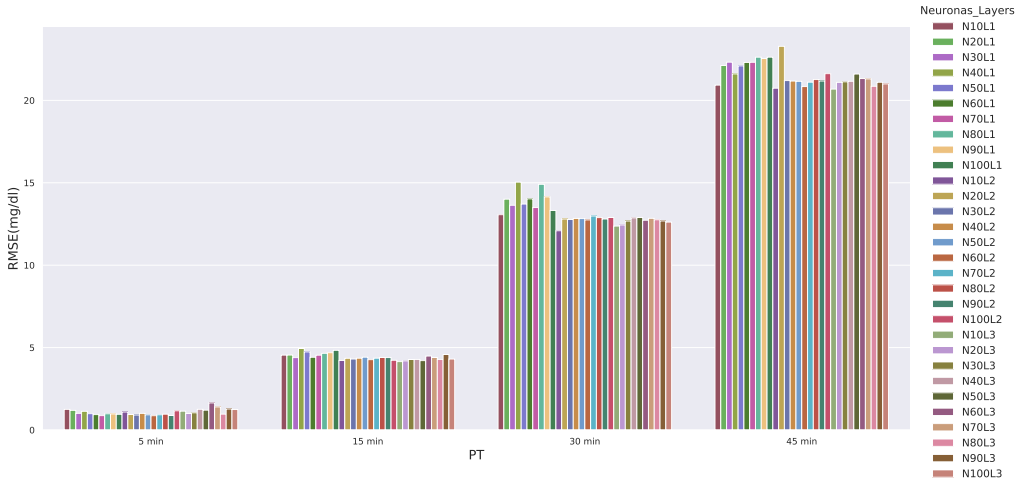
Figure 4.18: Box-plots for the e for the different architecture parameters for a input dimension of 50, being PT (a) 5 min (b) 15 min (c) 30 min (d) 45 min.

$$\begin{cases} m_w(r) = \beta_1 m_w(r-1) + (1+\beta_1)\alpha \frac{\partial L}{\partial W^m(r)} \\ v_w(r) = \beta_2 m_w(r-1) + (1+\beta_2)\alpha \left(\frac{\partial L}{\partial W^m(r)}\right)^2 \\ m_b(r) = \beta_1 m_b(r-1) + (1+\beta_1)\alpha \frac{\partial L}{\partial b^m(r)} \\ v_b(r) = \beta_2 m_b(r-1) + (1+\beta_2)\alpha \left(\frac{\partial L}{\partial b^m(r)}\right)^2 \end{cases} \tag{4.10}$$

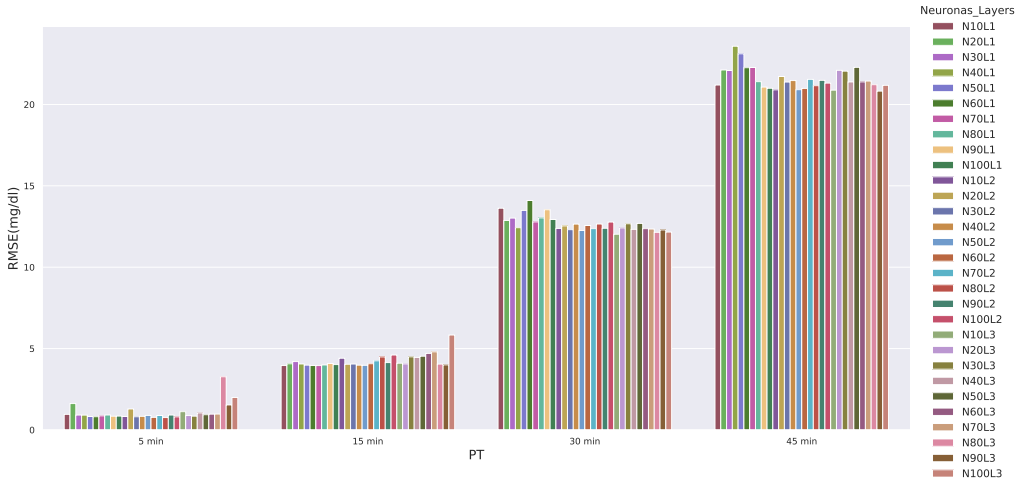
To update the weights and the biases, the unbiased estimators of the mean ($\hat{m}(r)$) and the variance ($\hat{v}(r)$) of the gradient are needed, see Equation 4.11.

$$\begin{cases} \hat{m}_w(r) = \frac{m_w(r)}{1-(\beta)^r} \\ \hat{v}_w(r) = \frac{v_w(r)}{1-(\beta)^r} \\ \hat{m}_b(r) = \frac{m_b(r)}{1-(\beta)^r} \\ \hat{v}_b(r) = \frac{v_b(r)}{1-(\beta)^r} \end{cases} \tag{4.11}$$

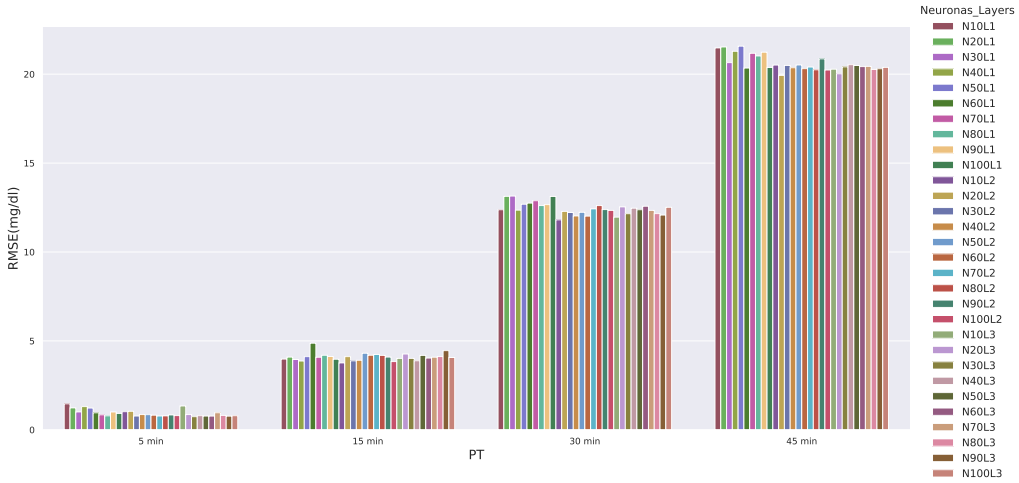
Once the previous steps are calculated, the weights and biases of the ANN are calculated as shown in the Equation 4.12. In this thesis, β_1 and β_2 have the values of 0.9 and 0.999, respectively, as it is stated by the authors .In addition, ϵ is fixed to 10^{-8}.



(a)



(b)



(c)

Figure 4.19: Chart graph with the RMSE for the different architecture parameters and grouped by the number of input dimension (a) 5, (b) 20 and (c) 50.

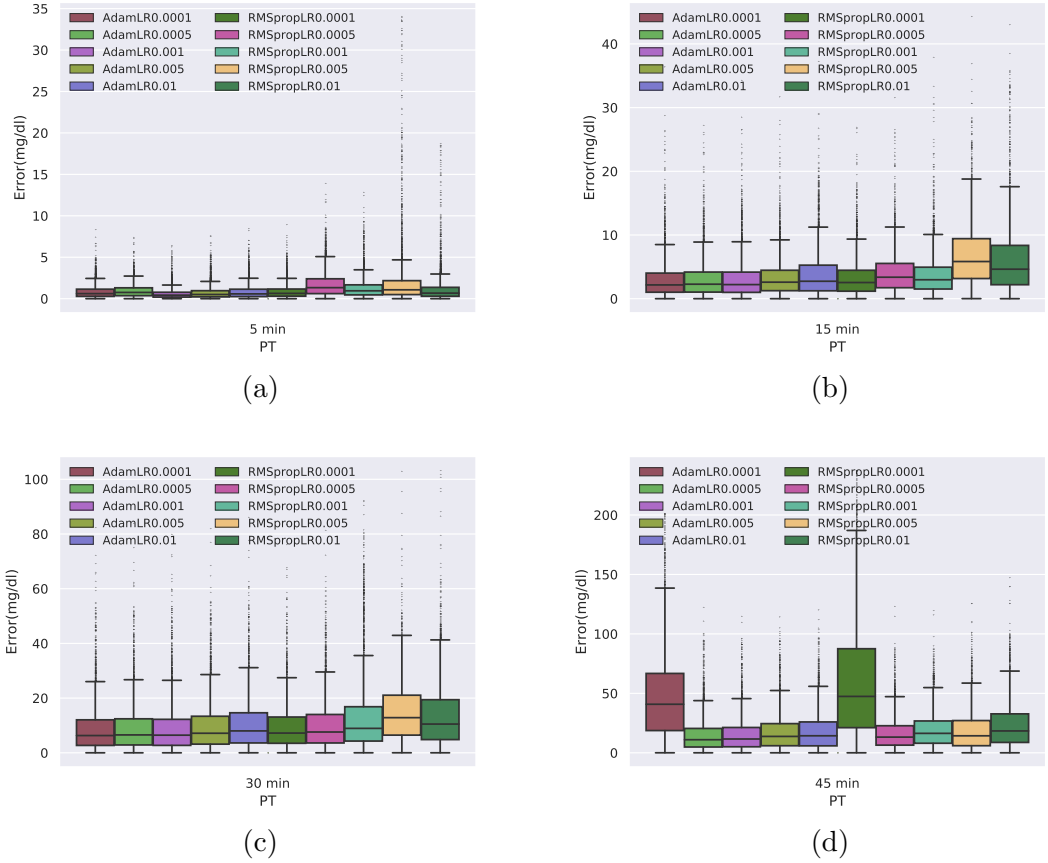


Figure 4.20: Box-plots for the e for the different training parameters for a input dimension of 5, being PT (a) 5 min (b) 15 min (c) 30 min (d) 45 min.

$$\begin{cases} W^{m+1}(r+1) = W^{m}(r) - \frac{\alpha}{\sqrt{\hat{v}_w(r)+\epsilon}}\hat{m}(r) \\ b^{m+1}(r+1) = b^{m}(r) - \frac{\alpha}{\sqrt{\hat{v}_b(r)+\epsilon}}\hat{m}(r) \end{cases} \tag{4.12}$$

The procedure is analogous to Section 4.4.1. The distribution error e is shown in Figures 4.20, 4.21 and 4.22 for the input dimension of 5, 20 and 50, respectively. In addition, in Figure 4.23 shows and the RMSE for each PT and each input dimension. Taken into account the results from both metrics, the best parameters are presented in Table 4.2 with all the optimal parameters. RMSprop has a good performance in many of the models with a few exceptions. However, The Adam optimizer has the best performance for all the models. The learning rates of 0,0001, 0.0005 and 0.001 are the ones that give the best performances. In total, twelve different models are selected where each one is specialized for a input dimension and a PT.

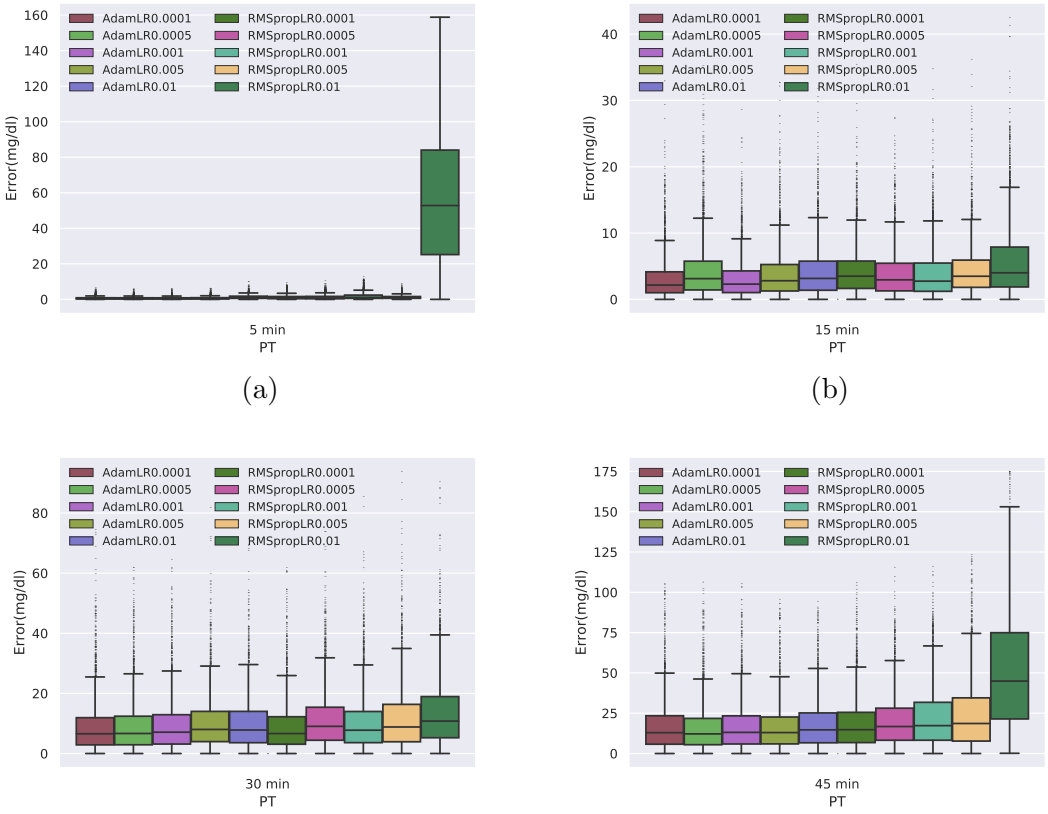


Figure 4.21: Box-plots for the e for the different training parameters for a input dimension of 20, being PT (a) 5 min (b) 15 min (c) 30 min (d) 45 min.

Input dimension	PT (min)	LSTM layers	Neurons	optimizer	α
5	5	2	60	Adam	0.001
5	15	3	10	Adam	0.001
5	30	2	10	Adam	0.0001
5	45	3	10	Adam	0.0005
20	5	2	80	Adam	0.0005
20	15	1	70	Adam	0.001
20	30	3	10	Adam	0.0001
20	45	3	90	Adam	0.0005
50	5	3	30	Adam	0.0001
50	15	2	10	Adam	0.0001
50	30	2	10	Adam	0.001
50	45	2	20	Adam	0.001

Table 4.2: Optimal parameters for each input dimension and PT. All these models have a Linear layer with 1 neuron in top of the last of the LSTM layer, see Section 4.3

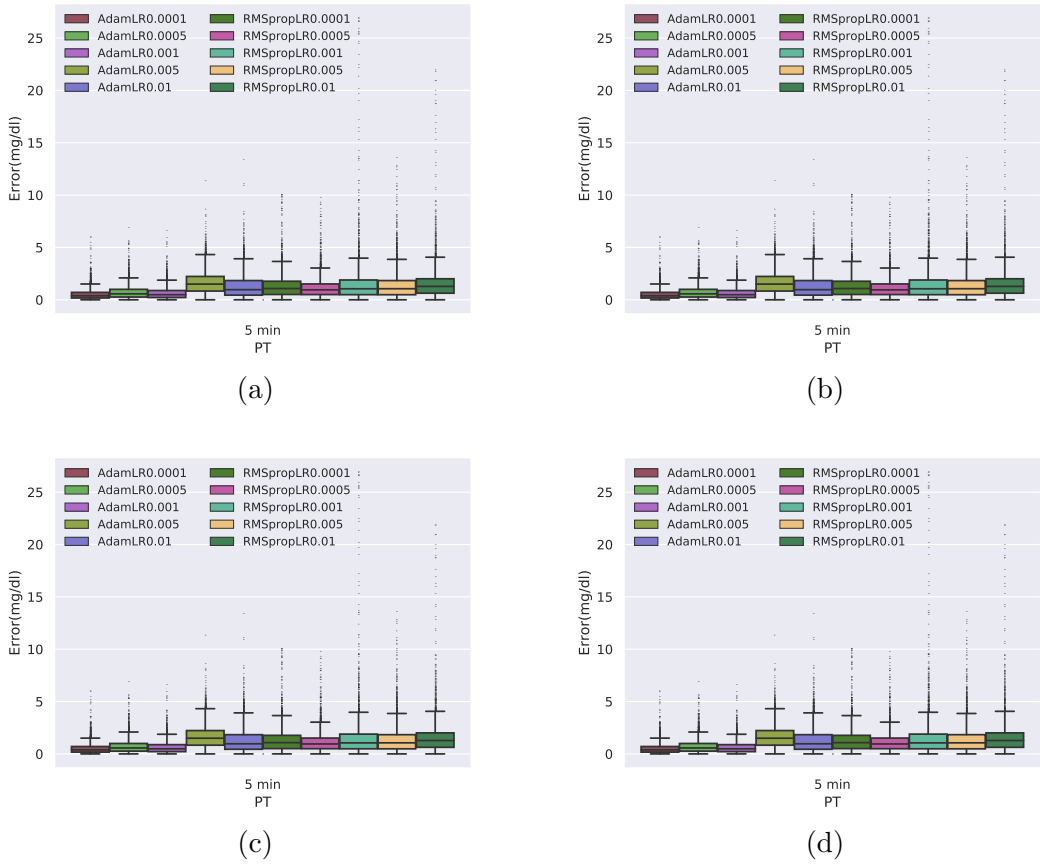
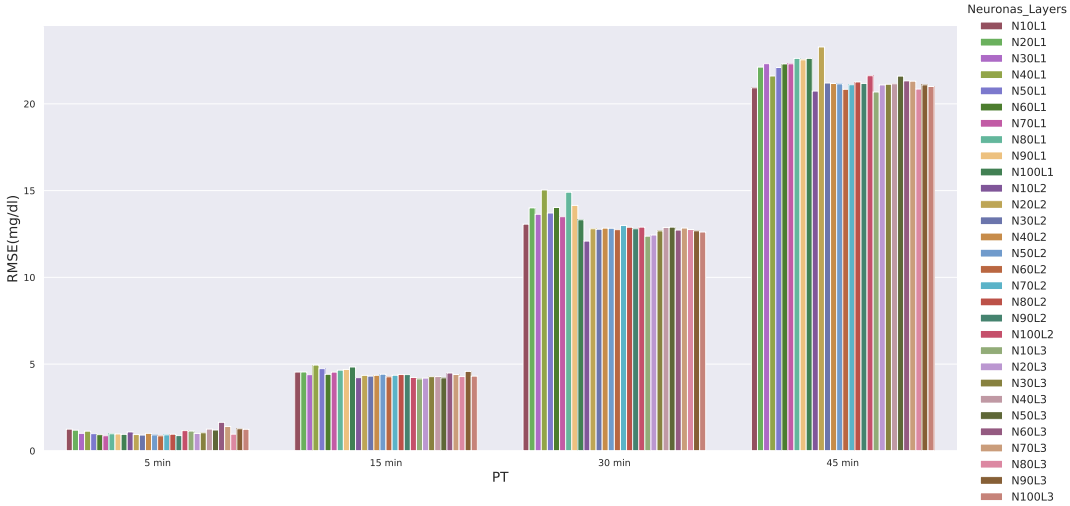
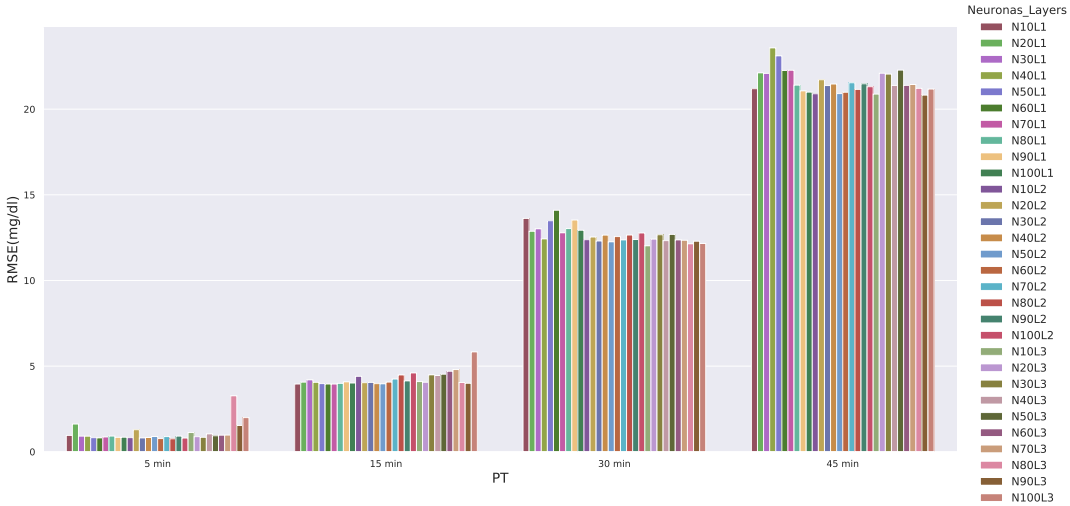


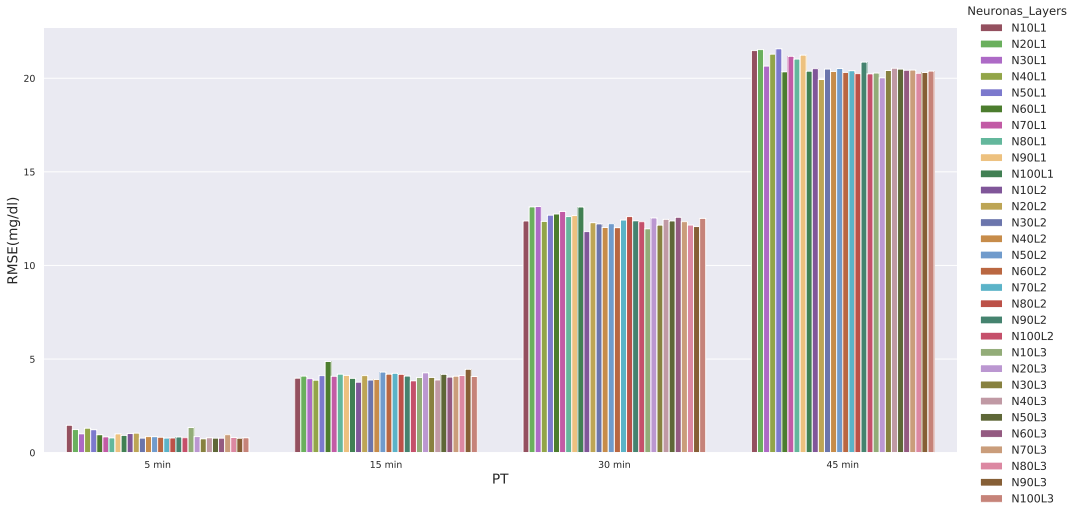
Figure 4.22: Box-plots for the e for the different training parameters for a input dimension of 50, being PT (a) 5 min (b) 15 min (c) 30 min (d) 45 min.



(a)



(b)



(c)

Figure 4.23: Chart graph with the RMSE for the different training parameters and grouped by the number of input dimension (a) 5, (b) 20 and (c) 50.

Chapter 5

Results

This chapter is devoted to present the evaluation metrics used to train and evaluate the predictors presented in Section 4.4. In addition, the results of these metrics and the performance of the models will be presented. To conclude, the results will be discused.

5.1 Metrics

In order to evaluate the performance of the predictors proposed in Section 4.4, as well as to define the loss function used in the BPTT algorithm to train the RNN, see Section 3.2.1, it is necessary to define a series of metrics. The metrics presented in this section are selected for their convenience to evaluate glucose prediction [48] and time series, in general [59]. The metrics stated in this Thesis measure the error made in the prediction and the correlation between the predicted signal and the targets. Another metric is presented that measures the delay associated to the prediction with respect to the original glucose signal. Let the prediction made by the proposed predictor be $Y = \{y_1, \ldots, y_n\}$, the correspondent expected output or targets $T = \{t_1, \ldots, t_n\}$, and N the number of samples to be evaluated.

First, let us define the loss function used in the back-propagation algorithm. The selected loss function is the mean squared error (MSE), see Equation 5.1. As the selected loss function MSE is mainly used for training the predictor.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(t_i - y_i)^2 \tag{5.1}$$

The following metrics are used in the evaluation of the performance of the proposed predictor. The first evaluation metric is the RMSE. It was already presented in Section 4.4 (see Equation 4.8). The RMSE, unlike the MSE, presents the error in the same unit that the variables that is evaluating. The RMSE provides an insight of the accuracy of the model predictions.

The next metric that will be discussed is the Pearson correlation coefficient ($r_{t,y}$) [48], see Equation 5.2. This coefficient gives information about the similarities between the target signal and the predicted signal.

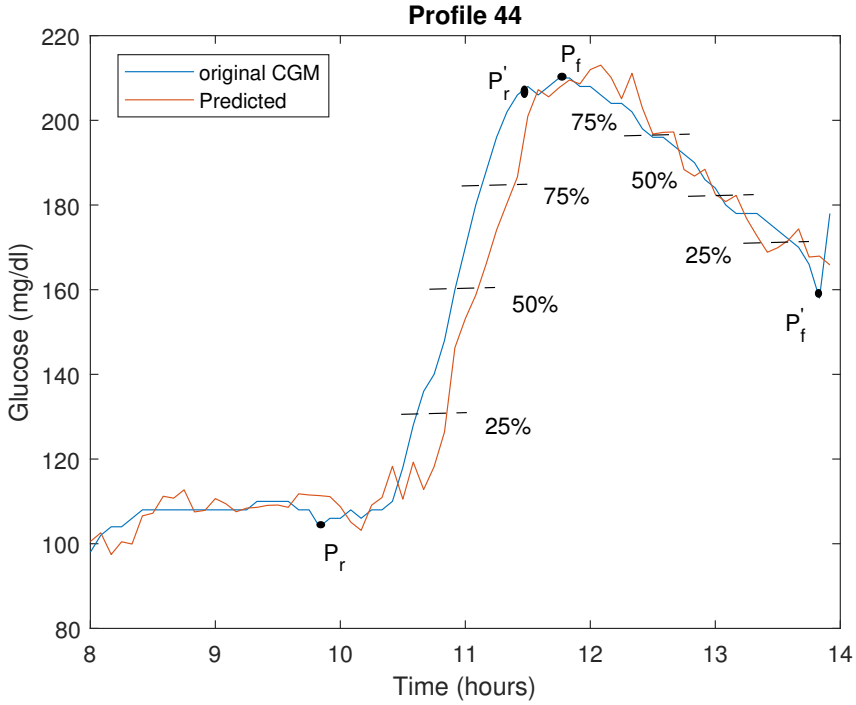


Figure 5.1: Example of the delay calculation, each percentage is calculate for the length of each slope. For the rising slope P_r is the lowest point and $P_r^{'}$ is the highest point. In the falling slope P_f is the highest point and $P_f^{'}$ is the lowest point.

$$r_{t,y} = \frac{\sum_{i=1}^{N}(t_i - \bar{t})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(t_i - \bar{t})^2}\sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}} \tag{5.2}$$

where $\bar{t}$ and $\bar{y}$ are the mean defined as $\bar{t} = \frac{1}{N}\sum_{i=1}^{n} t$ and $\bar{y} = \frac{1}{N}\sum_{i=1}^{n} y$.

The final metric is the calculation of the delay associated to the prediction. This metric was presented and developed by [49] and [48]. The purpose of this metric is to assess the lag produced in the prediction with respect to the original glucose signal from the CGM sensor. This metric measures the delay in the positive and negative slopes individually. Therefore, there are two values for this metric, one corresponds with the rising trend (upward delay) and the other with the falling trend (downward delay). The delay is calculated at 25%, 50% and 75% of the slope length. Being the slope length defined as the difference between the lowest point and the highest point. The upward and downward delay are finally computed as the average of all rising trends and all the falling trends, respectively. Figure 5.1 shows an example of how to calculate the delay.

As stated by the authors [49] to calculate the metric, it has been applied a first-order low-pass filter to the predicted signal and to the glucose original signal.

5.2 Results

In Section 4.4 twelve predictors were tuned to predict the glucose concentration by using three input parameters: glucose concentration from a CGM sensor, insulin, and

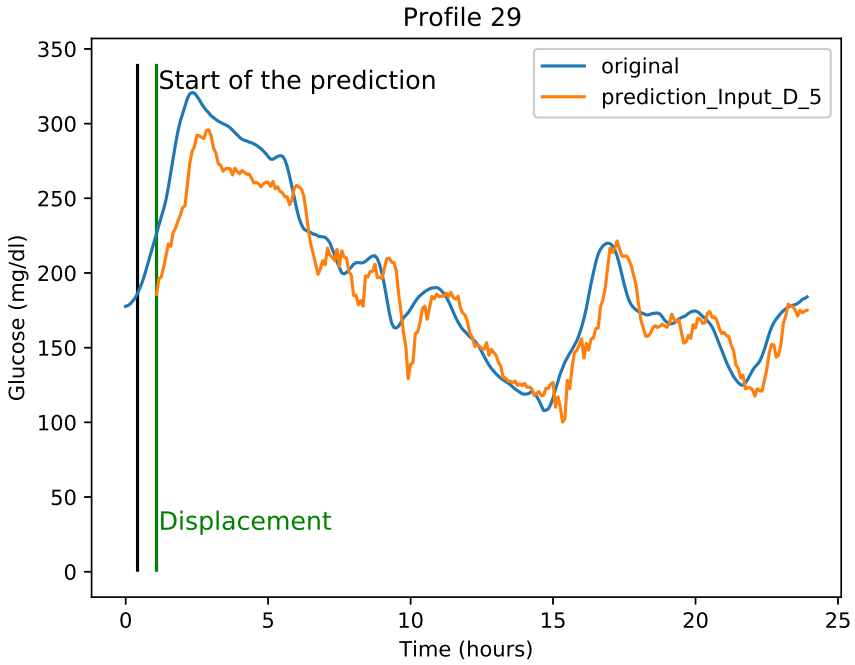


Figure 5.2: Example of the effect initialization time and the prediction in the illustration

the food intake. Each of the twelve predictors have an input dimension and a PT. There are three input dimensions: 5, 20 and 50. In addition, there are four PT: 5, 15, 30 and 45 minutes. The architecture and the parameters for each predictor are summarized in Table 4.2.

This study is divided in two parts. The first part consists of an evaluation of the accuracy of the 12 different predictors for their corresponding PT. After the models are evaluated individually, the second part assesses the effect of the prediction of using a different input dimension.

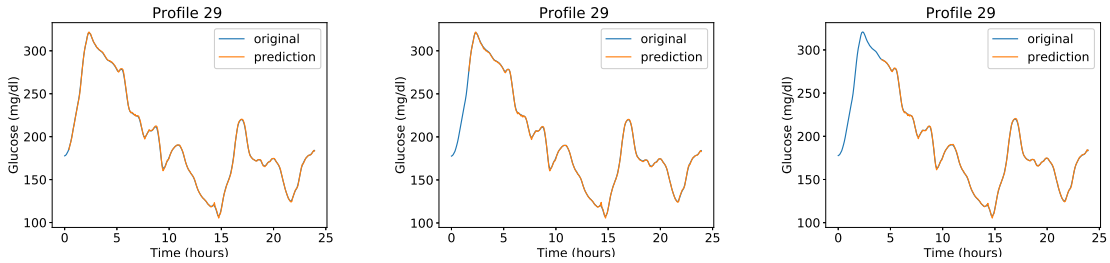
In the first half of the study, these twelve predictors are evaluated by the test dataset, see Section 4.2.2, and the metrics stated in Section 5.1. Moreover, to show that overfitting did not occur, the twelve predictors are also assessed with those metrics for the training dataset. The results from this metrics are displayed in Table 5.1 for the training dataset and in Table 5.2 for the evaluation dataset. In addition, the prediction for a profile of the test dataset for each model are displayed.

It is important to note that from this architecture, see Section 4.3, the prediction starts once that the first input dimension values are fed. An example of this is presented in Figure 5.2 with an input dimension of 5 and a PT of 45 min, the model will start to predict when receives the first 5 values. Therefore, being the samples separated by 5 minutes, the model of an input dimension of 5 will wait 25 minutes to begin the predictions. Once the model has done the first prediction, the model will forecast when a new input value is received.

First, it is important to verify that the models do not incur in overfitting. Table 5.1 and Table 5.2 show the results for the training and evaluation datasets, respectively. The RMSE is very similar for both datasets, being slightly lower for the training

Model	ID	PT (min)	RMSE (mg/dl)	$r_{t,y}$	U. delay (min)	D. delay (min)
60-60-1	5	5	1.03	0.99	0	0
10-10-10-1	5	15	4.4	0.99	5	5
10-10-1	5	30	12	0.96	15	10
10-10-10-1	5	45	20.06	0.9	30	15
80-80-1	20	5	0.9	0.99	0	0
70-1	20	15	4.39	0.99	5	0
10-10-10-1	20	30	11.97	0.96	15	5
90-90-90-1	20	45	19.6	0.9	25	10
30-30-30-1	50	5	0.86	0.99	0	0
10-10-1	50	15	3.66	0.99	5	0
10-10-1	50	30	11.59	0.96	15	5
20-20-1	50	45	17.94	0.90	25	10

Table 5.1: Metrics for all the models for the training dataset. The models are identified by the numbers of neurons in each layer and ID stands for Input dimension. U. delay and D. delay are defined as the upwards delay and the downward delay, respectively.



(a) (b) (c)

Figure 5.3: Predictions made a 5 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50

dataset. There are some exceptions where the evaluation dataset is slightly lower, these cases are the models for PT 5 and 15 minutes. The other metrics also have similar results for both datasets. Therefore, this is a good indicator that there has not been overfitting in our models.

A complete set of figures for the 12 combinations of predictors for profiles 29, 30, 48 and 50 can be found in Appendix C. The following figures will use a different profile for each prediction time for illustrative purposes.

There are three models specialized in the prediction of each PT. Firstly, let assess the performance of the models that predict in a PT of 5 min. Figure 5.3 shows the prediction made by the predictor with input dimension 5, 20 and 50, respectively. These three models have a high accuracy with error lower than 1 mg/dl, see Table 5.2, and the time series is almost identical. There is not delay for these models.

The performance of the models with a prediction time of 15 min is shown in Figure 5.4, for a predictor with input dimension 5, 20 and 50, respectively. These models improve the RMSE results presented in [49] for a prediction time of 15 min

Model	ID	PT (min)	RMSE (mg/dl)	$r_{t,y}$	U. delay (min)	D. delay (min)
60-60-1	5	5	0.96	0.99	0	0
10-10-10-1	5	15	4.34	0.99	5	5
10-10-1	5	30	12.6	0.96	15	10
10-10-10-1	5	45	21.40	0.89	30	15
80-80-1	20	5	0.86	0.99	0	0
70-1	20	15	4.23	0.99	5	0
10-10-10-1	20	30	12.51	0.96	15	5
90-90-90-1	20	45	21.52	0.89	25	20
30-30-30-1	50	5	0.85	0.99	0	0
10-10-1	50	15	3.67	0.99	5	0
10-10-1	50	30	12.20	0.96	15	10
20-20-1	50	45	20.76	0.88	25	15

Table 5.2: Metrics for all the models for the evaluation dataset. The models are identified by the numbers of neurons in each layer and ID stands for Input dimension. U. delay and D. delay are defined as the upwards delay and the downward delay, respectively.

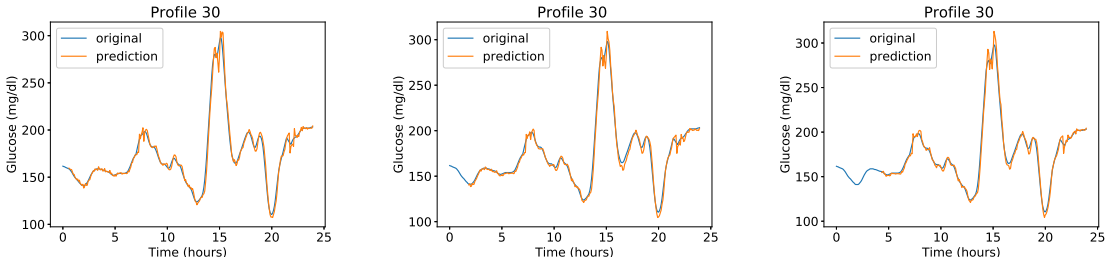


Figure 5.4: Predictions made a 15 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50

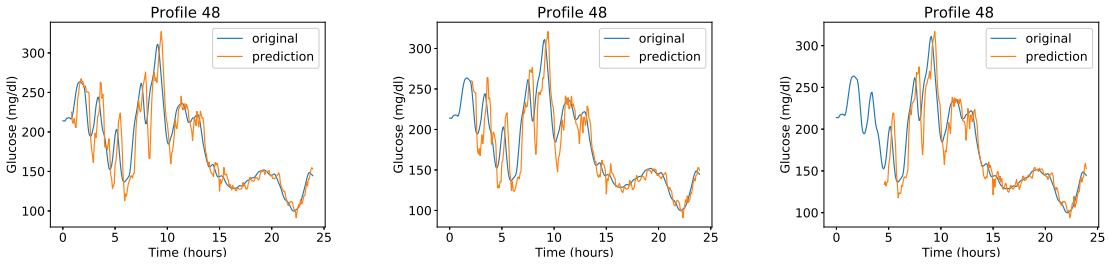
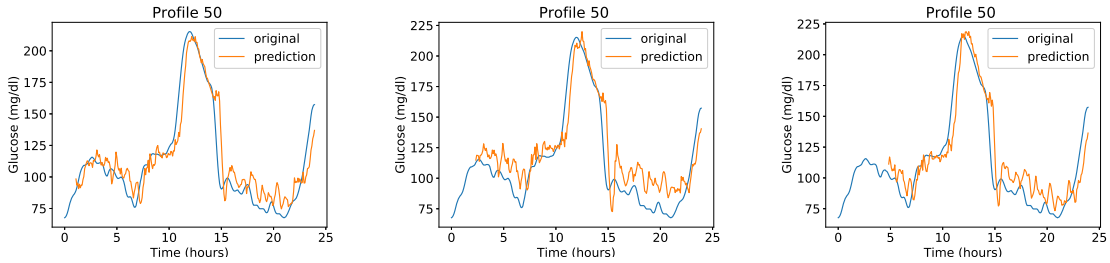


Figure 5.5: Predictions made a 30 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



(a) (b) (c)

Figure 5.6: Predictions made a 45 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50

in a 55.44%, 56.57% and 62.32%, respectively. Moreover, Figure 5.4 shows that the predicted glucose signals resemble with the original values from the CGM sensor. This idea is reinforced by the value of the correlation, $r_{t,y}$, that is very close to 1 (see Table 5.2). Furthermore, the models present a delay of 5 minutes in the upward slopes. However, the model is faster in the downward slopes, which explains the models for input dimensions 20 and 50 do not have this kind of delay.

The glucose predictions made for a time of 30 min are shown in Figure 5.5 for a predictor with input dimension 5, 20 and 50, respectively. Comparing the results with those obtained in [49] for a PT of 30 min, these predictors improve the RMSE in a 27.79%, 28.31% and 30.09%, respectively. To conclude, the $r_{t,y}$ is very close to 1 (see Table 5.2), saying that the predicted signal is very close to the original from the Guardian® Real Time. The delays are similar for all the models that predict for that PT.

Lastly, the models with a PT of 45 min are shown in Figure 5.6. The results from these predictors, see Table 5.2, improves the RMSE obtained in [49] by a 17.54%, 16.98% and 20.53%, respectively. The similarity between the predicted signal and the original from the CGM sensor have worsened with regard to the previous cases where the $r_{t,y}$ was very close to 1. To conclude, the presented models are faster in the falling slopes than in the upward slopes.

After the study of the performance of all the predictors, the most accurate models are the ones with an input dimension of 50 (see Table 5.2). Moreover, these models are the faster in both rising and falling slopes. This result was as expected because it is the model that receives more information in their inputs. However, the evaluation metrics in Table 5.2 present very similar performance between models with the same PT, independently of the input dimension.

Figures 5.7, 5.8, 5.9 and 5.10 also present a high similitude between the prediction at the same PT. Some differences start to appear when the PT is 30 min or 45 min between the performance of the model. Specifically, the differences are better appreciated when there is an steep slope. In these cases, the predictors with input dimensions of 5 and 20 tend to overshoot a bit more than the predictors with input dimensions of 50. Nevertheless, these differences are difficult to appreciate and are not very pronounced.

Another issue that was previously commented, but it is more noticeable in Figures 5.7, 5.8, 5.9 and 5.10, it is that depending of the input dimension each

predictor has a time of initialization to start the predictions. This initialization time is longer if the input dimension is bigger.

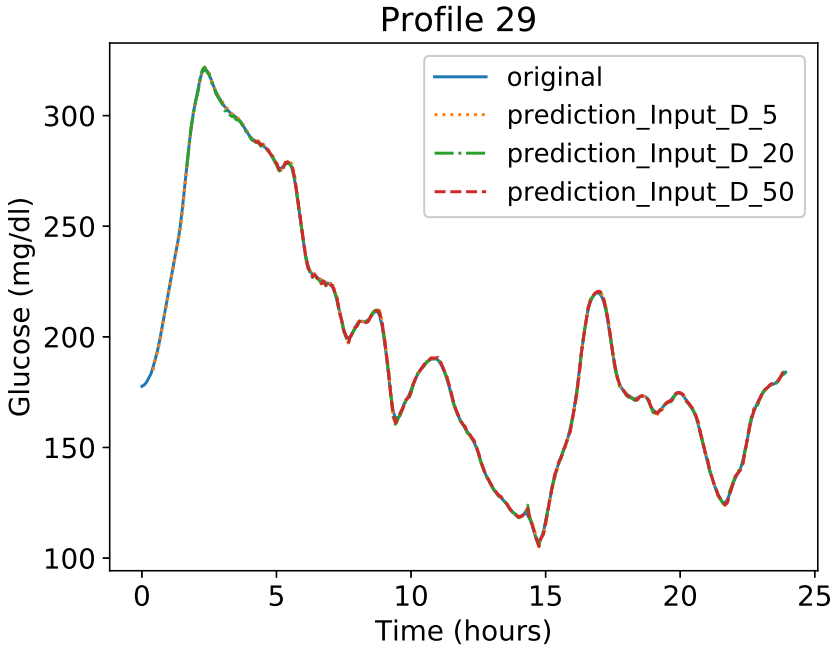


Figure 5.7: Predictions made a 5 minutes by the models with an input dimension of 5, 20 and 50

5.3 Discussion

Twelve models have been deployed to achieve the objective of predicting glucose concentration in patients with diabetes type 1. These models may be grouped by their PT, being then four groups in function of their prediction time: 5 min, 15 min, 30 min and 45 min.

However, these predictors are practical if they give enough time to the patient to make the necessary modification to his or her treatment [49]. Consequently, the predictors with a PT of 5 min are not very suitable for this task, even if they are the most accurate models.

Predictors specialized to forecast at 15 min are still short of time to be practical in usual terms. Although, these models may be useful in some cases due to their high accuracy. A prediction time of 30 minutes or longer is wide enough to allow a patient to do the necessary changes to prevent the occurrences of hypoglycemias or hyperglycemias. Predictors with a PT of both 30 and 45 minutes will proportionate a good range of time of action, being both useful to predict the glucose concentration. However, predictors with a PT of 30 minutes have the best compromise between the range of time that they provide to the patient to change their treatment and the accuracy of the predictions.

In Section 5.2, the performance of the predictors with the same PT and different

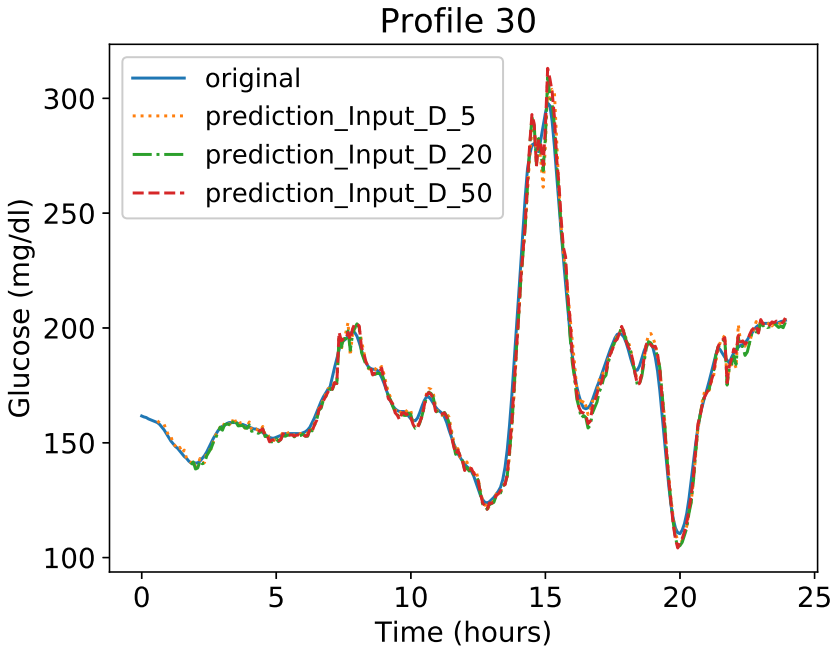


Figure 5.8: Predictions made a 15 minutes by the models with an input dimension of 5, 20 and 50

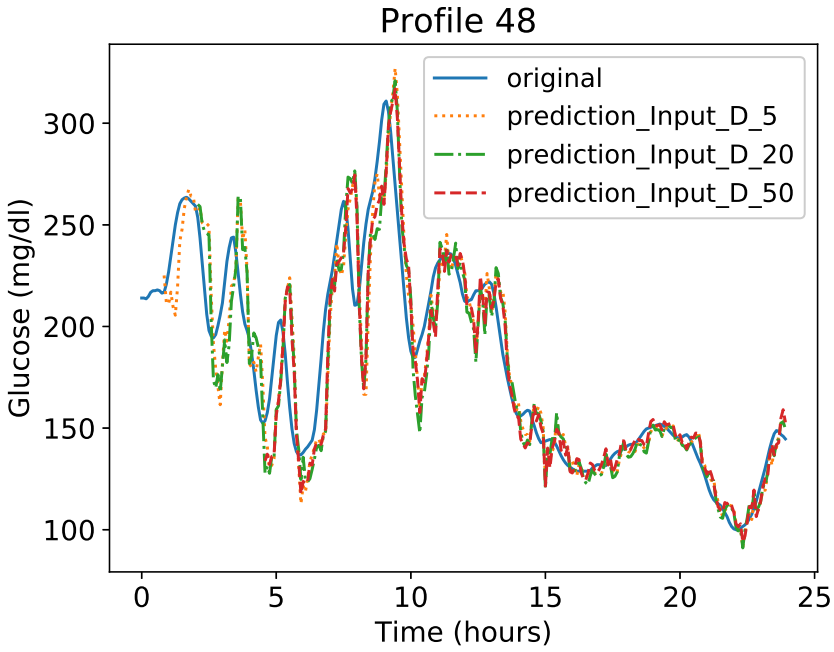


Figure 5.9: Predictions made a 30 minutes by the models with an input dimension of 5, 20 and 50

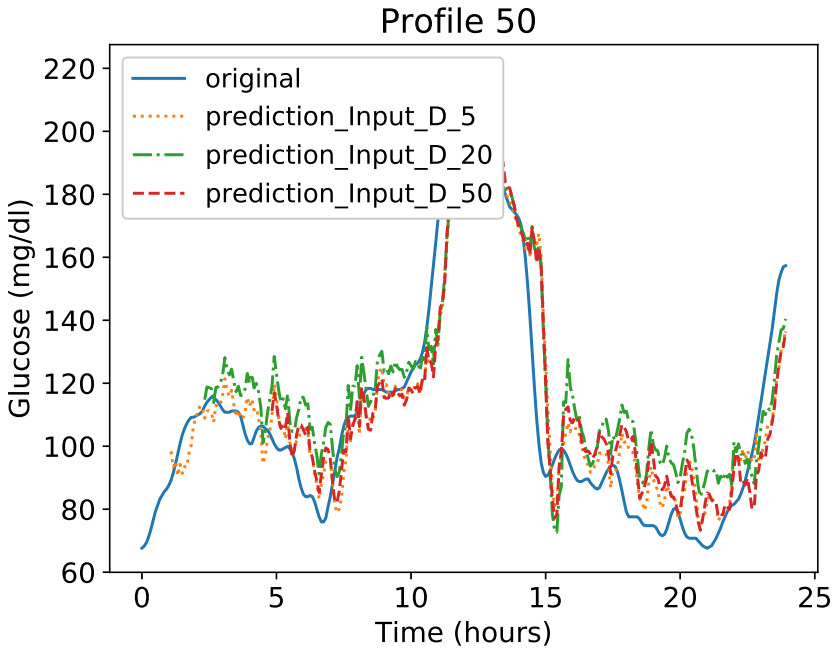


Figure 5.10: Predictions made a 45 minutes by the models with an input dimension of 5, 20 and 50

input dimension was compared, assessing the effect of the input dimension in the predictor's output. It is important to consider all the characteristics of the predictors before to choose one. These characteristics are the input dimension, the prediction time (PT) and the results of the metrics evaluation (see Table 5.2). Taking all this into account, the results from Table 5.2 are very similar to each predictor at the same PT, however the predictors with bigger input dimension have a longer initialization time. Therefore, having all the models a similar accuracy, it would be more likely to choose the models with less input dimensions because they starts to predict sooner.

Nevertheless, the most accurate predictors were those with an input dimension of 50. This happen as expected because they are the models which receive the more information. However, the results from the others models are very similar. The improvement of adding more information in the input is small. This effect can be explained for several reasons.

First, there are not enough data for the model to learn the long term dependencies. In the training dataset (see Section 4.2.2), there are 36 profiles with 288 samples each one. This is a small dataset taking into account that LSTMs have at least four more weights and biases (see Section 3.2.2) than a multilayer perceptron. This is translated in LSTM needing far more data for training.

Another reason that there is not enough data is based off of the nature of the condition of type 1 diabetes patients (see Chapter 2). Patients with diabetes type one does not segregate insulin, hence, they do not have the mechanisms that regulates the glucose concentration. So when the glucose starts to increase the value, it does it without control and same happens when it decreases. The LSTM neural network are fed by the glucose concentration. The predictor could give priority to the recent values in time because they have a bigger impact in the future values of the glucose.

Hence, the most importance characteristic of the glucose is the tendency where the predictor tends to ignore the oldest values that have a lower impact in the tendency of the signal.

The mechanism that patients have to change the glucose concentration are mainly the injection of insulin, which diminishes the glucose value, and the ingestion of carbohydrates, that increases the glucose concentration. There are others parameters that affects the concentration of glucose, but those parameters are not fed to the predictors developed in this Thesis. However, the effects of both insulin and ingestion of carbohydrates are not immediate. This arises another possible problems. Insulin start to acts one hour after its injection and the ingestion of the carbohydrates could take hours too. Therefore, these are the long term dependencies that the predictor should learn to improve the prediction.

The problem states that the bolus of insulin and the food intake is presented a few times in each profile, 3 o 4 times on average that corresponds with main meals. Then, considering that the training dataset only has 36 profiles. Hence there is not enough data so the predictor can properly learn the effects of these long term dependencies in the future glucose values. Therefore, the oldest input values are being partially ignored.

Moreover, they may be long term dependencies in the glucose concentration parameter. These dependencies might manifest over longer periods of time as events that happen daily at similar hours, for example, the resistance to the insulin in the morning. However, in the current dataset, each profile has been arranged as independent days. Therefore, these long term dependencies could not be learnt by the neural network. Future research with profiles longer than 1 day would be needed to assess the impact of this long term dependencies.

Another reason is that, there are more parameters that have effect in the glucose concentration such as the physical exercise. For example, physical exercise reduces the glucose levels, but it also has effects after the execution (see Section 2.3.3). These other parameters could be producing unexpected effects that might result in a worsening of the model performance. However, a future research is needed to test the hypotheses stated here.

Nevertheless, the twelve models presented in Section 5.2 are more accurate than the obtained in [49], for each prediction time 15, 30 and 45 minutes, respectively. In addition, the twelve models presented are slower in the upwards trend than the model presented in [49]. However, the twelve models are much faster in the downwards trends than the obtained in [49]. Actually, this may be a desirable characteristic because diabetic patients are usually more worry about the hypoglycemias than the hyperglycemias.

Therefore, the LSTM based architecture presented in this Thesis represents an improvement in the RMSE and downward slope delay metrics, over the multilayer perceptron stated in the previous study of [49]. Therefore, neural networks based on LSTMs are appropriate for the prediction of glucose concentration in diabetes type 1. However, an increment in the size of the dataset could improve the accuracy of the LSTM based predictor and remains for future work.

Chapter 6

Conclusions and future lines

The aims of this chapter is to present the more important conclusions and results of this Thesis, as well as to propose future lines of research based in it.

6.1 Conclusions

The main problem of diabetes are the complications caused by the occurrences of hyperglycemias and hypoglycemias. Therefore, the main diabetes' treatment is to control the glucose concentration to be between the normal values. This task is complicated because the number of parameters that influences the blood glucose concentration, see Section 2.3. A predictor could improve the control of the disease allowing patients to prevent hypoglycemias and hyperglycemias.

The aim of this Thesis was to continue the work presented in previous studies [49] with the design of LSTM based predictors. The evaluation of the predictors was done at four different prediction times (PT) 5, 15, 30 and 45 minutes and the predictors were fed with three different input dimensions 5, 20 and 50. To achieve these tasks, a main architecture based on LSTMs was developed to create the basic model of the future predictors. From this architecture twelve predictors were deployed, each one specialized in a PT with a specific input dimension. *PyTorch* was the selected framework to implement the designed predictors.

The predictors with PT of 5 minutes and 15 minutes were the most accurate models. However, these prediction times are too short for a patient to prevent a hypoglycemia or a hyperglycemia, but they may have others uses where the accuracy is much more important than a long prediction time.

Predictors with a PT of both 30 and 45 minutes provide enough time to do the necessary modifications in the treatment to prevent the occurrences of hypoglycemias or hyperglycemias, being both PT practical in usual terms. Nevertheless, predictors with a PT of 30 minutes provides the best compromise between the amount of time that provides the patient to modify his or her treatment and the performance of the model predictions.

Comparing the models with the same PT and different input dimension, the most accurate and with best performance was the model with an input dimension of 50. However, the models with an input dimension of 5 and 20 also had a similar performance to the model of input dimension of 50. Therefore, it would be more recommendable to use an input dimension of 5 because it has a much shorter

initialization time.

Taking all this into consideration, the most suitable predictor may be the predictor with a input dimension of 5 and a prediction time of 30. Although the rest of the models may be also used depending of the necessity of prediction that want to be covered.

The twelve predictors proposed present a good accuracy in the performance and a suitable fit to the original glucose to be predicted. In addition, these predictors surpasses the model presented in [49] as was the main objective stated in this Thesis. The results obtained are encouraging and support the use of LSTM for predictions of glucose concentration in patients with diabetes type 1.

6.2 Future lines

The work presented in this Thesis opens new lines of research and future works to improve the results obtained by the predictors, some of them are introduced:

- The use of Convolutional Neural Networks in tandem with the architecture presented in this Thesis could enhance the performance of the predictors.
- In this Thesis only three parameters are used to calculate the futures values of glucose, these are the past values of glucose provided by a CGM sensor, the insulin injected by the patients and the food intake. However, the process that regulates the glucose concentration is a complex mechanism and other parameters such as physical exercise have an impact. Therefore, another future research is to add new parameters to feed the predictors and assess the importance of these parameters in the performance of the predictor.
- The process of tuning the parameters in Section 4.4 may be done using novel ideas such as Bayesian optimization [61] or using AutoML, an example is presented in [18].
- The glucose from the CGM sensor have several artifacts, see Section 4.2.1.1. In this Thesis, a manual approach has been followed to correct them. However, a more novel approach may be followed using an ANN that performs this task. This ANN could correct the distortions caused by the calibrations, the missing samples and filter the signal.
- To conclude, some causes has been proposed to explain why the different input dimension gives such a similar results, see Section 5.3. The most probable cause is the necessity of more data to train the predictor based on LSTMs. Therefore, two approaches may be followed: first, it is the acquisition of more data from real patients. However, this is a very difficult task because there are a lot of requirements to fulfill and patients may not follow the instructions. Moreover, this approach is very expensive. The second approach consists of using the paradigm of few-shot learning, where the models are trained with small amounts of data, [32] and [67] present works in this direction.

Bibliography

[1] K. G. Alberti and P. Z. Zimmet. Definition, diagnosis and classification of diabetes mellitus and its complications. Part 1: diagnosis and classification of diabetes mellitus provisional report of a WHO consultation. *Diabetic Medicine: A Journal of the British Diabetic Association*, 15(7):539–553, July 1998.

[2] F. Allam, Z. Nossai, H. Gomma, I. Ibrahim, and M. Abdelsalam. A Recurrent Neural Network Approach for Predicting Glucose Concentration in Type-1 Diabetic Patients. In L. Iliadis and C. Jayne, editors, *Engineering Applications of Neural Networks*, IFIP Advances in Information and Communication Technology, pages 254–259. Springer Berlin Heidelberg, 2011.

[3] American Diabetes Association. Standards of Medical Care in Diabetes—2013. *Diabetes Care*, 36(Supplement 1):S11–S66, Jan. 2013.

[4] American Diabetes Association. 6. Glycemic Targets: Standards of Medical Care in Diabetes—2019. *Diabetes Care*, 42(Supplement 1):S61–S70, Jan. 2019.

[5] American Diabetes Association. 7. Diabetes Technology: Standards of Medical Care in Diabetes—2019. *Diabetes Care*, 42(Supplement 1):S71–S80, Jan. 2019.

[6] American Diabetes Association. Diabetes Care: 42 (Supplement 1). *Diabetes Care*, 42(Supplement 1), Jan. 2019.

[7] Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *arXiv:1206.5538 [cs]*, June 2012. arXiv: 1206.5538.

[8] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, Mar. 1994.

[9] E. M. Benjamin. Self-Monitoring of Blood Glucose: The Basics. *Clinical Diabetes*, 20(1):45–47, Jan. 2002.

[10] J. M. Berg, J. L. Tymoczko, and L. Stryer. Each Organ Has a Unique Metabolic Profile. *Biochemistry. 5th edition*, 2002.

[11] Canadian Diabetes Association Clinical Practice Guidelines Expert Committee and A. Y. Y. Cheng. Canadian Diabetes Association 2013 clinical practice guidelines for the prevention and management of diabetes in Canada. Introduction. *Canadian Journal of Diabetes*, 37 Suppl 1:S1–3, Apr. 2013.

[12] J. Ceruelo Bermejo, R. Miranda Hidalgo, and A. García Ortiz. Insulinas: clasificación y usos. *Scyl ITE. Boletín De Información Terapéutica*, 1, 2005.

[13] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *arXiv:1409.1259 [cs, stat]*, Sept. 2014. arXiv: 1409.1259.

[14] Clínica Universidad de Navarra. ¿Qué es insulinemia? `https://www.cun.es/diccionario-medico/terminos/insulinemia`. [Accessed May 27, 2019].

[15] C. Crespo, M. Brosa, A. Soria-Juan, A. Lopez-Alba, N. López-Martínez, and B. Soria. Costes directos de la diabetes mellitus y de sus complicaciones en España (Estudio SECCAID: Spain estimated cost Ciberdem-Cabimer in Diabetes). *Avances en Diabetología*, 29(6):182–189, Nov. 2013.

[16] H. B. Demuth, M. H. Beale, O. De Jesus, and M. T. Hagan. *Neural Network Design*. Martin Hagan, USA, 2nd edition, 2014.

[17] I. D. Federation. *IDF Diabetes Atlas*. Brussels, Belgium, eighth edition edition, 2017.

[18] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter. Efficient and Robust Automated Machine Learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2962–2970. Curran Associates, Inc., 2015.

[19] S. K. Garg, H. K. Hoff, and H. P. Chase. The role of continuous glucose sensors in diabetes care. *Endocrinology and Metabolism Clinics of North America*, 33(1):163–173, Mar. 2004.

[20] Geoffrey Hinton, N Srivastava, and Kevin Swersky. Neural Networks for Machine Learning - Lecture 6a - Overview of mini-batch gradient descent. `https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf`, 2012. [Accessed May 30, 2019].

[21] L. Heinemann. Finger Pricking and Pain: A Never Ending Story. *Journal of Diabetes Science and Technology*, 2(5):919–921, Sept. 2008.

[22] S. Hochreiter. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 06(02):107–116, Apr. 1998.

[23] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997.

[24] J. D. Hunter. Matplotlib: A 2d Graphics Environment. *Computing in Science & Engineering*, 9(3):90–95, May 2007.

[25] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[26] John E. Hall. *Guyton and Hall Textbook of Medical Physiology.* Elsevier, Philadelphia, PA, 13th edition edition, 2016.

[27] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado, M. Hughes, and J. Dean. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, Dec. 2017.

[28] Juvenile Diabetes Research Foundation Continuous Glucose Monitoring Study Group, W. V. Tamborlane, R. W. Beck, B. W. Bode, B. Buckingham, H. P. Chase, R. Clemons, R. Fiallo-Scharer, L. A. Fox, L. K. Gilliam, I. B. Hirsch, E. S. Huang, C. Kollman, A. J. Kowalski, L. Laffel, J. M. Lawrence, J. Lee, N. Mauras, M. O'Grady, K. J. Ruedy, M. Tansey, E. Tsalikian, S. Weinzimer, D. M. Wilson, H. Wolpert, T. Wysocki, and D. Xing. Continuous glucose monitoring and intensive treatment of type 1 diabetes. *The New England Journal of Medicine*, 359(14):1464–1476, Oct. 2008.

[29] P. Kapoor and S. S. Bedi. Weather Forecasting Using Sliding Window Algorithm. *ISRN Signal Processing*, 2013:1–5, 2013.

[30] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, Dec. 2014. arXiv: 1412.6980.

[31] V. Kotu and B. Deshpande. Chapter 12 - Time Series Forecasting. In V. Kotu and B. Deshpande, editors, *Data Science (Second Edition)*, pages 395–445. Morgan Kaufmann, Jan. 2019.

[32] Z. Li, F. Zhou, F. Chen, and H. Li. Meta-SGD: Learning to Learn Quickly for Few-Shot Learning. *arXiv:1707.09835 [cs]*, July 2017. arXiv: 1707.09835.

[33] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. W. M. van der Laak, B. van Ginneken, and C. I. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60–88, Dec. 2017.

[34] J. Lowe, S. Linjawi, M. Mensch, K. James, and J. Attia. Flexible eating and flexible insulin dosing in patients with diabetes: Results of an intensive self-management course. *Diabetes Research and Clinical Practice*, 80(3):439–443, June 2008.

[35] C. R. Marling and R. C. Bunescu. The OhioT1dm Dataset For Blood Glucose Level Prediction. In *KHD@IJCAI*, 2018.

[36] MATLAB. *version 7.10.0 (R2010a).* The MathWorks Inc, Natick, Massachusetts, 2010.

[37] W. McKinney. Data Structures for Statistical Computing in Python. In S. van der Walt and J. Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51–56, 2010.

[38] Medtronic. Medical Technology, Services, and Solutions Global Leader. https://www.medtronic.com/us-en/index.html. [Accessed May 27, 2019].

[39] Medtronic. MiniMed 670g Insulin Pump System | Medtronic Diabetes. https://www.medtronicdiabetes.com/products/minimed-670g-insulin-pump-system, 2018. [Accessed May 27, 2019].

[40] Medtronic. Guardian Sensor 3 | Medtronic Diabetes. https://www.medtronicdiabetes.com/products/guardian-sensor-3, 2019. [Accessed May 27, 2019].

[41] J. M. Nazzal, I. M. El-Emary, and S. A. Najim. Multilayer Perceptron Neural Network (MLPs) For Analyzing the Properties of Jordan Oil Shale. *World Applied Sciences Journal*, page 7, 2008.

[42] C. Olah. Understanding LSTM Networks – colah's blog. http://colah.github.io/posts/2015-08-Understanding-LSTMs/, Aug. 2015. [Accessed May 15, 2019].

[43] T. E. Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[44] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. *NIPS-W*, Oct. 2017.

[45] A. Pfützner, C. Schipper, M. Niemeyer, M. Qvist, A. Löffler, T. Forst, and P. B. Musholt. Comparison of patient preference for two insulin injection pen devices in relation to patient dexterity skills. *Journal of Diabetes Science and Technology*, 6(4):910–916, July 2012.

[46] L. Prechelt. Early Stopping - But When? pages 55–69, 1998.

[47] H. Purwins, B. Li, T. Virtanen, J. Schlüter, S.-y. Chang, and T. Sainath. Deep Learning for Audio Signal Processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):206–219, May 2019. arXiv: 1905.00078.

[48] C. Pérez Gandía. *Propuesta de algoritmos de predicción de glucosa en pacientes diabéticos*. phd, E.T.S.I. Telecomunicación (UPM), Mar. 2014.

[49] C. Pérez-Gandía, A. Facchinetti, G. Sparacino, C. Cobelli, E. J. Gómez, M. Rigla, A. de Leiva, and M. E. Hernando. Artificial neural network algorithm for online glucose prediction from continuous glucose monitoring. *Diabetes Technology & Therapeutics*, 12(1):81–88, Jan. 2010.

[50] Roche Diagnostics, S. L. Accu-Chek Aviva. https://www.accu-chek.es/sistemas-de-medicion/aviva, 2015. [Accessed May 27, 2019].

[51] G. Roglic, editor. *Global report on diabetes*. World Health Organization, Geneva, Switzerland, 2016. OCLC: ocn948336981.

[52] R. Rojas. The Backpropagation Algorithm. In *Neural Networks*, pages 149–182. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.

[53] F. Rosenblatt. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, 65(6):386–408, 1958.

[54] L. Rui. Energy Metabolism in the Liver. *Comprehensive Physiology*, 4(1):177–197, Jan. 2014.

[55] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533, Oct. 1986.

[56] sanofi-aventis US LLC. Lantus SoloStar Injection Guide. `https://www.lantus.com/-/media/EMS/Conditions/Diabetes/Brands/Lantus2/Consumer/desktop/PDF/Lantus-SoloStar-Pen-Guide.pdf?la=en`, 2019. [Accessed May 27, 2019].

[57] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, Jan. 2015.

[58] R. J. Schrot. Targeting Plasma Glucose: Preprandial Versus Postprandial. *Clinical Diabetes*, 22(4):169–172, Oct. 2004.

[59] R. Sendra Arranz. *Design and implementation of an HVAC consumption prediction system based on LSTM neural networks*. PhD thesis, Universidad Politécnica de Madrid, Madrid, 2018.

[60] SingHealth. Insulin syringe preparation. `https://www.healthxchange.sg/diabetes/essential-guide-diabetes/insulin-syringe-preparation-how-mix-short-intermediate-acting-insulin`, 2016. [Accessed May 27, 2019].

[61] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. *arXiv:1206.2944 [cs, stat]*, June 2012. arXiv: 1206.2944.

[62] G. Sparacino, F. Zanderigo, S. Corazza, A. Maran, A. Facchinetti, and C. Cobelli. Glucose concentration can be predicted ahead in time from continuous glucose monitoring sensor time-series. *IEEE transactions on bio-medical engineering*, 54(5):931–937, May 2007.

[63] S. A. Tabish. Is Diabetes Becoming the Biggest Epidemic of the Twenty-first Century? *International Journal of Health Sciences*, 1(2):V, July 2007.

[64] N. X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, pages 1–8, Montreal, Quebec, Canada, 2009. ACM Press.

[65] G. Williams and J. C. Pickup. *The Handbook of Diabetes*. Wiley-Blackwell, Malden, Mass, 3rd edition edition, Mar. 2004.

[66] H.-C. Yeh, T. T. Brown, N. Maruthur, P. Ranasinghe, Z. Berger, Y. D. Suh, L. M. Wilson, E. B. Haberl, J. Brick, E. B. Bass, and S. H. Golden. Comparative

effectiveness and safety of methods of insulin delivery and glucose monitoring for diabetes mellitus: a systematic review and meta-analysis. *Annals of Internal Medicine*, 157(5):336–347, Sept. 2012.

[67] E. Zakharov, A. Shysheya, E. Burkov, and V. Lempitsky. Few-Shot Adversarial Learning of Realistic Neural Talking Head Models. *arXiv:1905.08233 [cs]*, May 2019. arXiv: 1905.08233.

[68] T. Zhu, K. Li, P. Herrero, J. Chen, and P. Georgiou. A Deep Learning Algorithm For Personalized Blood Glucose Prediction. In *KHD@IJCAI*, 2018.

Appendix A

Impact

Diabetes is an important health care problem. According to a global report by World Health Organization (WHO) [51], around 422 million adults, globally, had diabetes in 2014. Diabetes is one the main expenditures of the national healthcare system, most of the expenditure are caused by the complications associated to the disease [15]. The main complications of this disease comes from the occurrences of hypoglycemias and hyperglycemias. These complications could be avoided or reduced with control of the blood glucose levels. This project aims to help achieve this goal. Therefore, the impact of this project strives for a reduction of diabetes complications due to a better control of the blood glucose concentration.

- **Social impact:** this project will have a direct impact on patients with diabetes type 1 and also to their relatives. Patients may prevent the diabetes complications, avoiding diseases that lead to high dependency such as heart attacks, strokes, kidney failure, blindness, etc. This results in a better quality of life for patients and their relatives.
- **Economical impact:** this project will have a direct economical impact for patient and for the national healthcare system. The expenses of the national healthcare system would be reduced due to the decrease of medical emergencies and the treatment of the diabetes complications. Moreover, patients will reduce the expenditure of the treatments in the diseases caused by diabetes complications.
- **Environmental impact:** this project will also have an environmental impact. Thanks to a better control of the blood glucose concentration, the medical waste related with the treatment of diabetes complications will be reduced.

Appendix B

Budget

This project has been developed during four months in the Escuela Técnica Superior de Ingenieros de Telecomunicación from the Universidad Politécnica de Madrid using some of its resources. An approximate budget was calculated by taking into consideration of the cost of human resources, and software and technical equipment:

- **Costs derived from Human resources**:

  This section of the budget considers the salaries of the staff involved in this project: project manager (engineer) and the engineer student, author of this Thesis, as shown on Table B.1.

	Cost per hour (€)	**Working hours**	**Total cost (€)**
Project manager	22	85	1870
Engineering student	15	600	9000
TOTAL			**10870**

Table B.1: Costs derived from Human resources

- **Costs derived from Software and technical equipment**:

  This section of the budget takes into account the software and technical equipment used in the development of this Thesis, see Table B.2. The total cost has been calculated by the product of the depreciation cost per month and the time of use.

	Lifetime (years)	**Units**	**Cost (€)**	**Depreciation (€/months)**	**Time used (months)**	**Total cost (€)**
GTX 1080 Ti	4	1	1000	20.83	4	83.32
Personal computer	5	1	800	13.33	4	53.32
MATLAB License	1	1	2000	166.66	4	664
TOTAL						**803.49**

Table B.2: Costs derived from Software and technical equipment

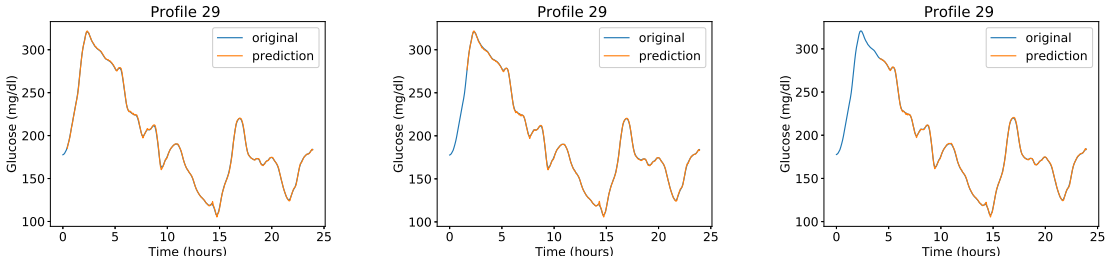
Considering both parts of the budget, the total cost of this Thesis amounts to **116873.3**.

Appendix C

Additional results

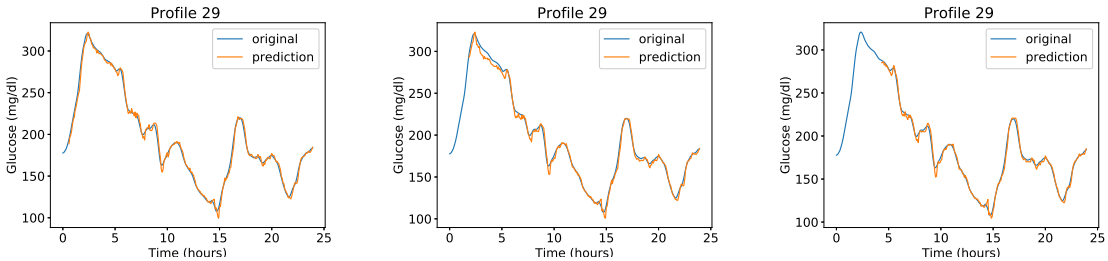
This appendix shows the performance of the twelve predictors for the profiles 29, 30, 48 and 50. This is a continuation of the figures shown in the Chapter 5.

The results for the twelve predictors for the profile 29 are shown in the Figures C.1, C.2, C.3 and C.4. The results of applying all the input dimensions in a same PT are shown in Figure C.5.



(a) (b) (c)

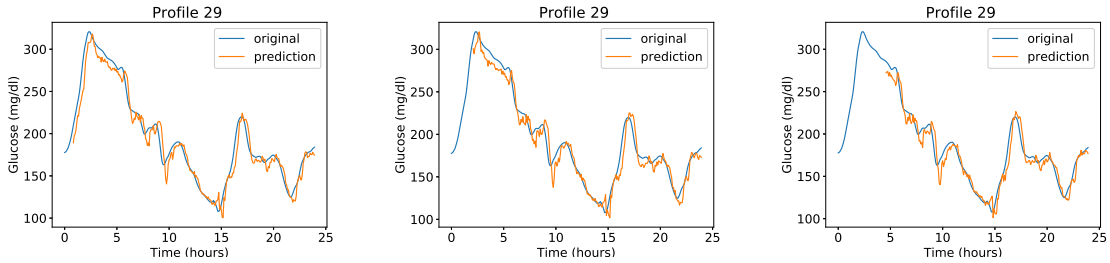
Figure C.1: Predictions for profile 29 made a 5 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



(a) (b) (c)

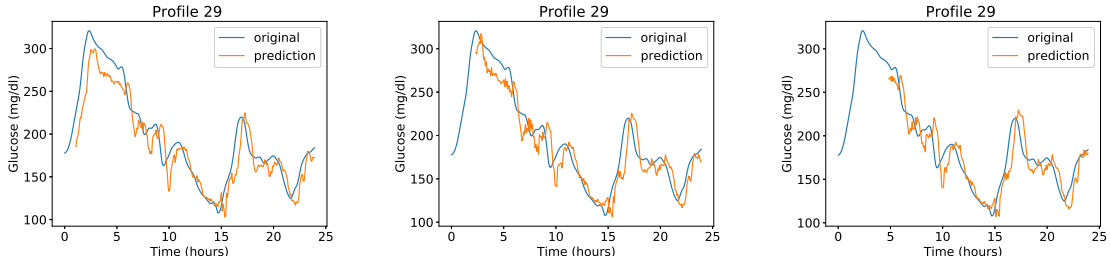
Figure C.2: Predictions for profile 29 made a 15 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50

Figures C.6, C.7, C.8 and C.9 show the prediction made by the 12 models. Figure C.10 show the results of applying all the input dimensions in a same PT.



(a) (b) (c)

Figure C.3: Predictions for profile 29 made a 30 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



(a) (b) (c)

Figure C.4: Predictions for profile 29 made a 45 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50

The results for the twelve predictors for the profile 48 are shown in the Figures C.11, C.12, C.13 and C.14. The results of applying all the input dimensions in a same PT are shown in Figure C.15.

Figures C.16, C.17, C.18 and C.19 show the prediction made by the 12 models. Figure C.20 show the results of applying all the input dimensions in a same PT.

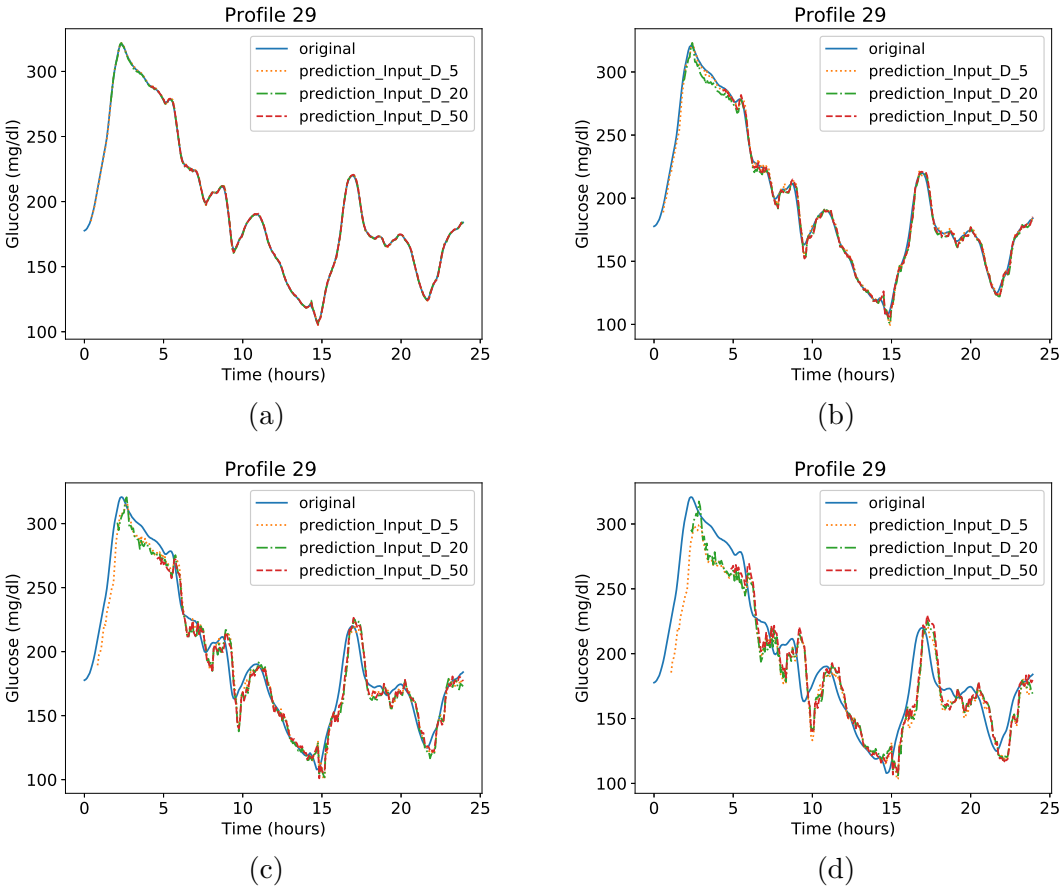


Figure C.5: Predictions for profile 29 with the three dimension inputs (5, 20 and 50) and a PT of: (a) 5 min, (b) 15 min, (c) 30 min and (d) 45 min

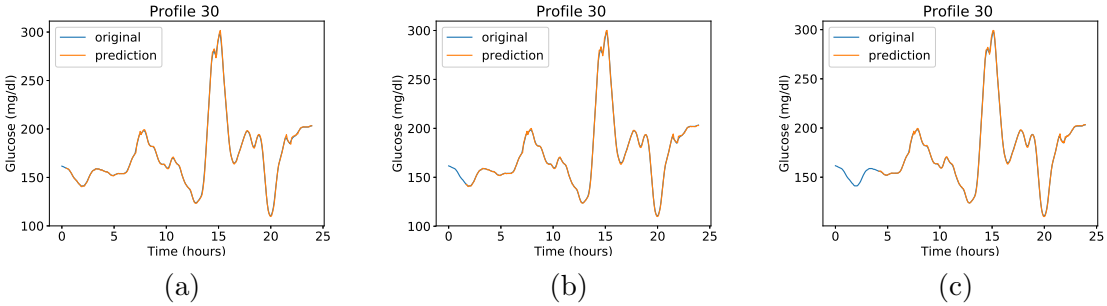
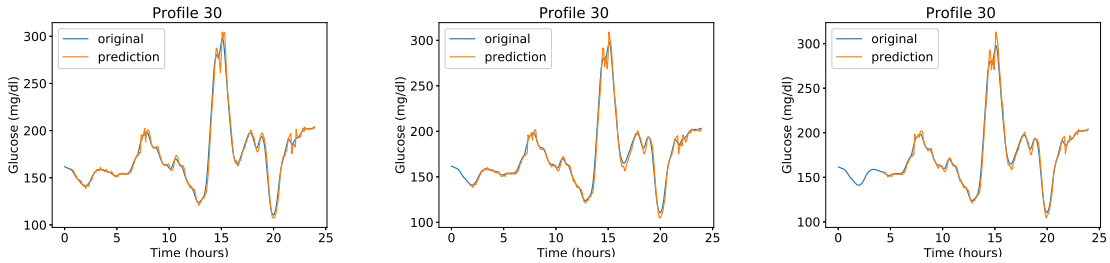
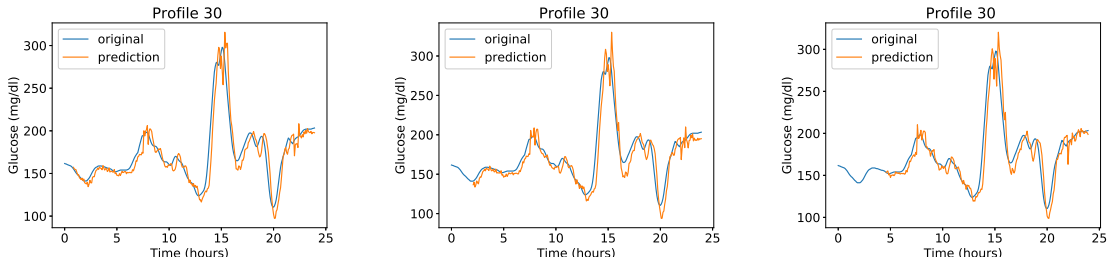


Figure C.6: Predictions for profile 30 made a 5 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



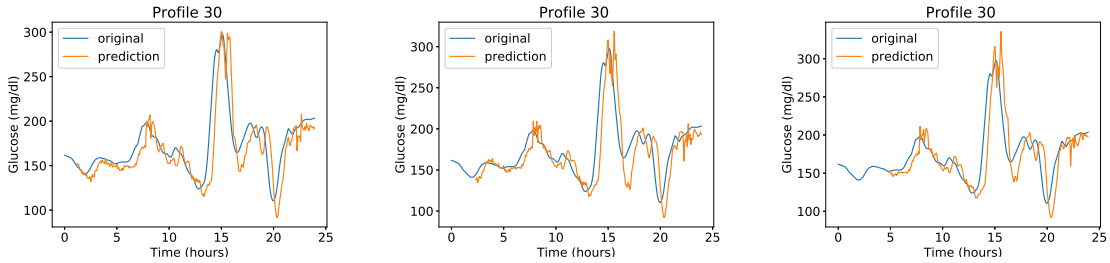
(a) (b) (c)

Figure C.7: Predictions for profile 30 made a 15 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



(a) (b) (c)

Figure C.8: Predictions for profile 30 made a 30 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



(a) (b) (c)

Figure C.9: Predictions for profile 30 made a 45 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50

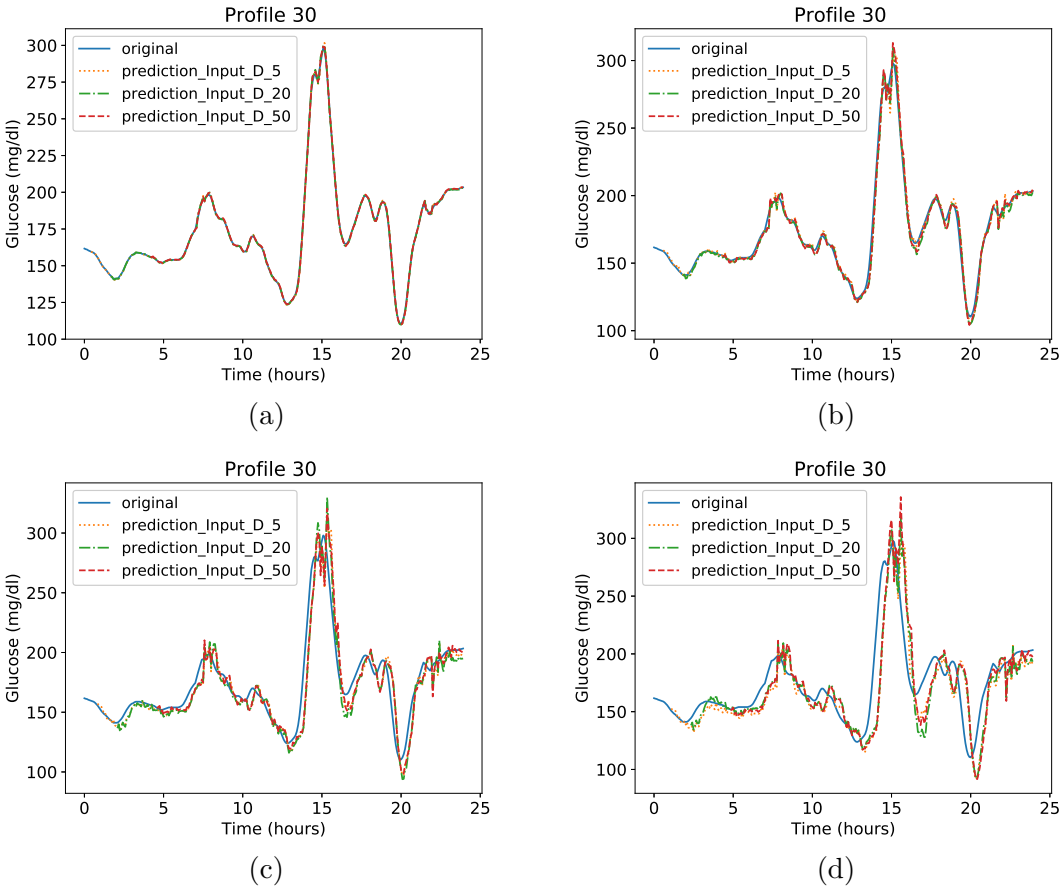


Figure C.10: Predictions for profile 30 with the three dimension inputs (5, 20 and 50) and a PT of: (a) 5 min, (b) 15 min, (c) 30 min and (d) 45 min

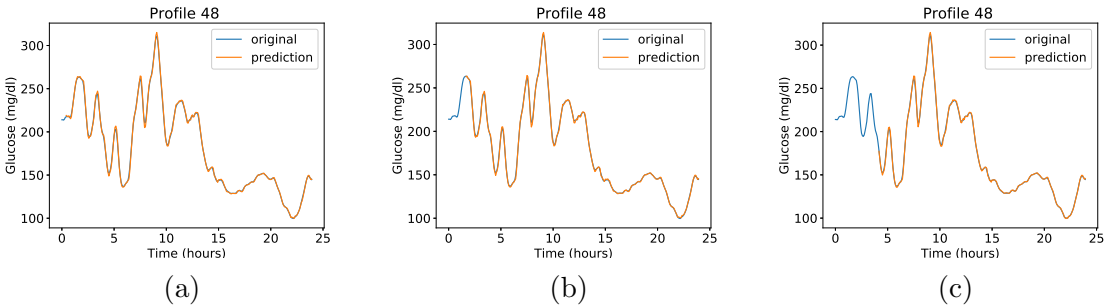
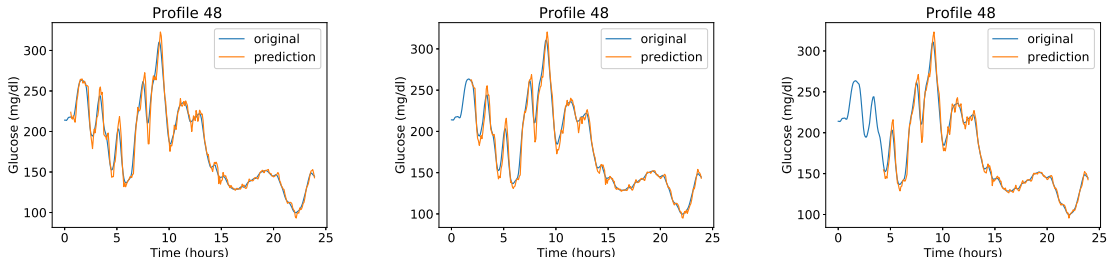
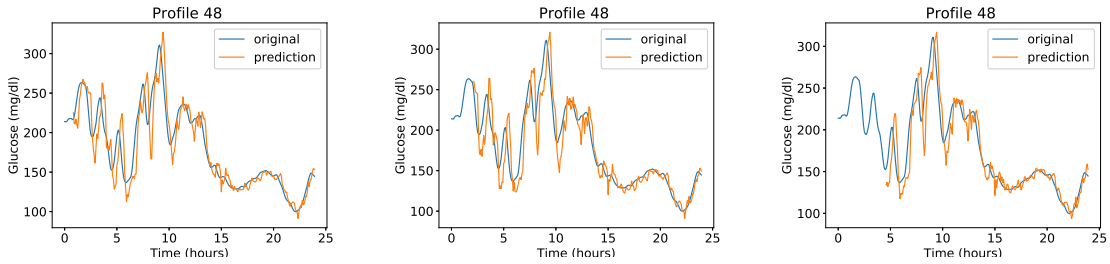


Figure C.11: Predictions for profile 48 made a 5 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



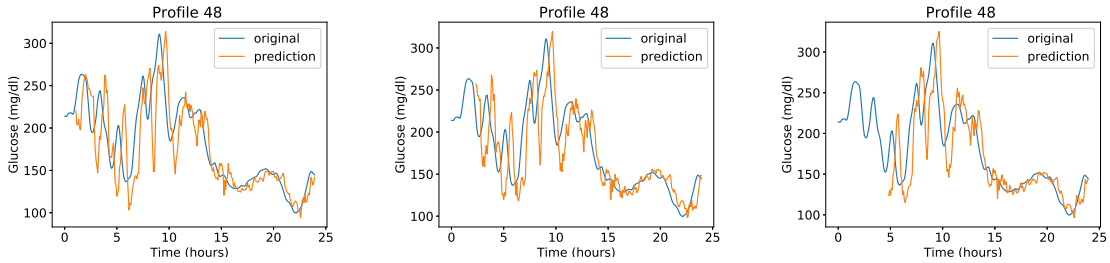
(a) (b) (c)

Figure C.12: Predictions for profile 48 made a 15 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



(a) (b) (c)

Figure C.13: Predictions for profile 48 made a 30 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



(a) (b) (c)

Figure C.14: Predictions for profile 48 made a 45 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50

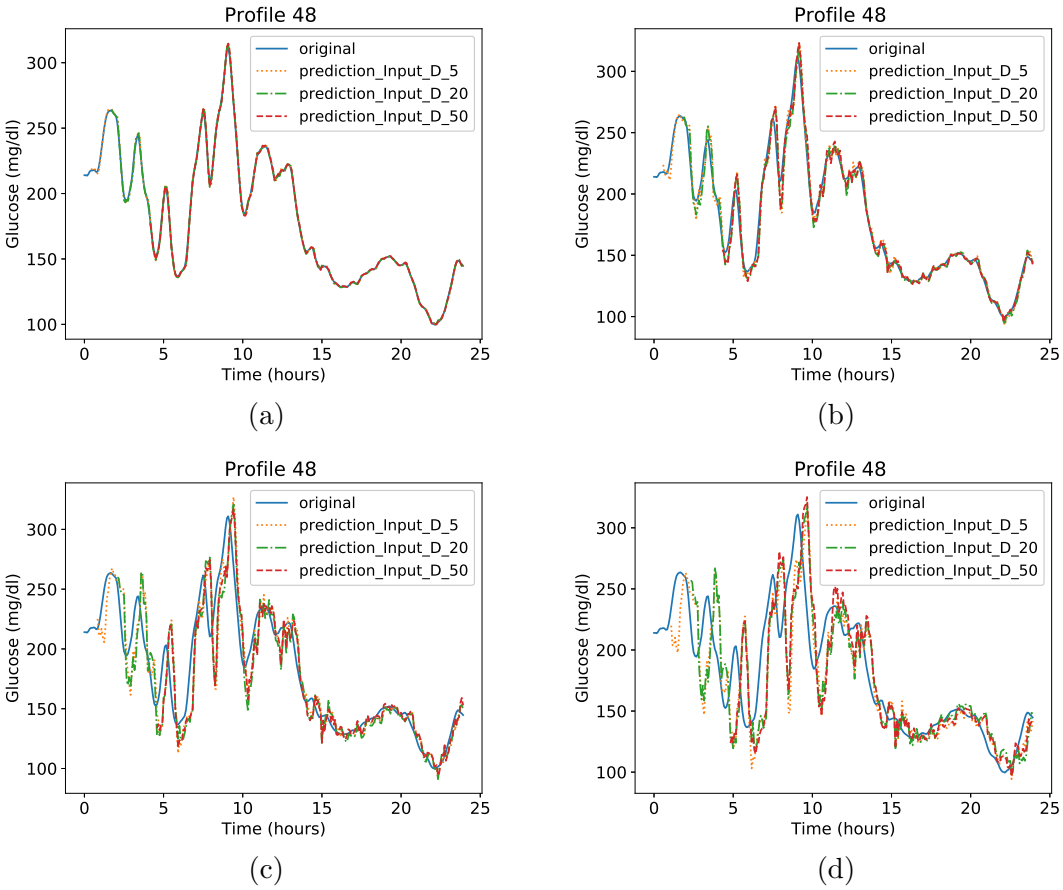


Figure C.15: Predictions for profile 48 with the three dimension inputs (5, 20 and 50) and a PT of: (a) 5 min, (b) 15 min, (c) 30 min and (d) 45 min

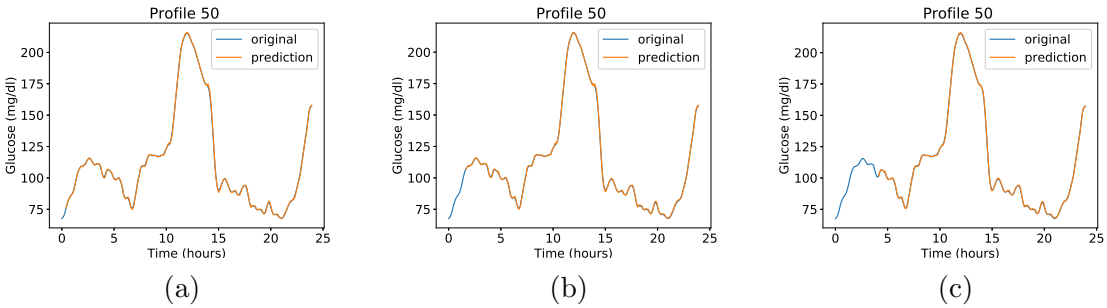
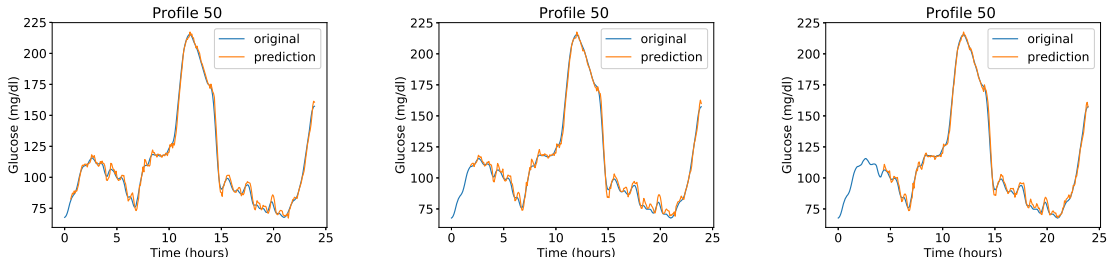
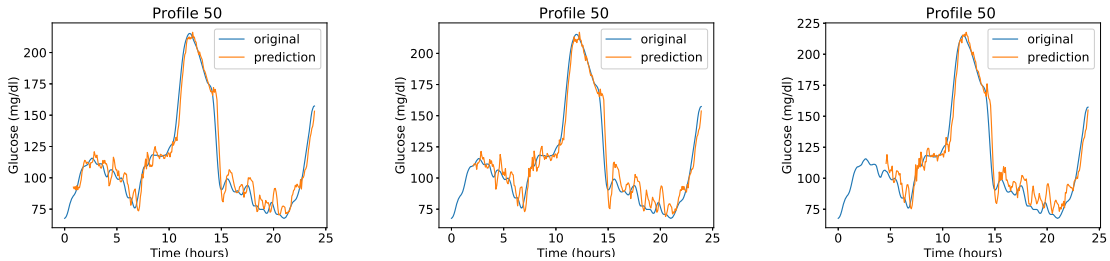


Figure C.16: Predictions for profile 50 made a 5 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



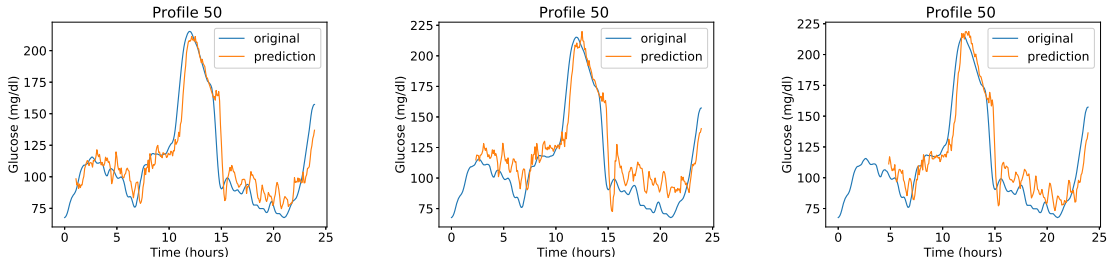
(a) (b) (c)

Figure C.17: Predictions for profile 50 made a 15 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



(a) (b) (c)

Figure C.18: Predictions for profile 50 made a 30 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50



(a) (b) (c)

Figure C.19: Predictions for profile 50 made a 45 minutes by the model with an input dimension of: (a) 5 , (b) 20 and (c) 50

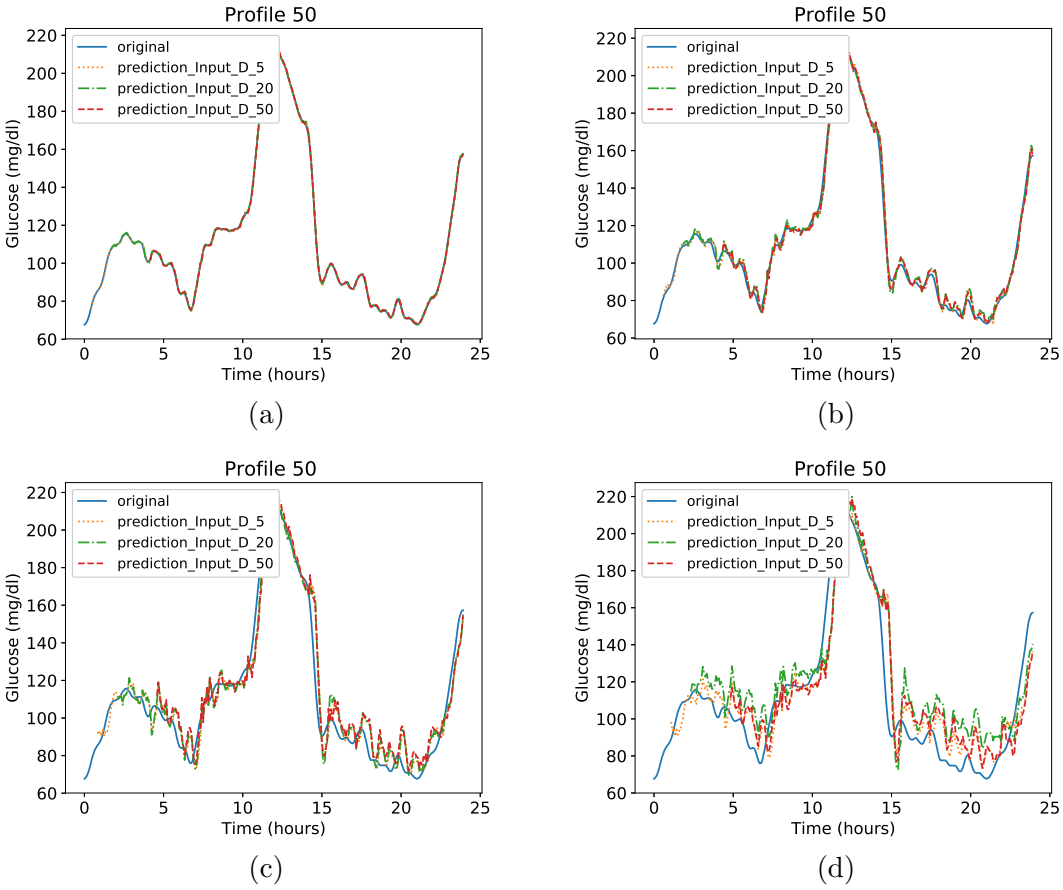


Figure C.20: Predictions for profile 50 with the three dimension inputs (5, 20 and 50) and a PT of: (a) 5 min, (b) 15 min, (c) 30 min and (d) 45 min