

Universidad Complutense de Madrid

FACULTAD DE INFORMATICA

GLUCOSE CLASSIFICATION AND
PREDICTION SYSTEM WITH
NEURAL NETWORKS

Final Degree Project

Authors:

Alejandro Varela Lorenzo

Alvaro Delgado Gutierrez

Advisor: J. Ignacio Hidalgo

June 2020

Contents

1	Summary	3
2	Introduction	5
2.1	Background	5
2.2	Goals	7
2.2.1	Development of glucose prediction models	7
2.2.2	Website service	7
2.3	Work-plan	7
3	State of the art	9
4	GlucNet	15
4.1	Devices	15
4.2	Data sources	16
4.3	Data preprocessing	17
4.3.1	Study of the data-set	17
4.3.2	Data filling	18
4.3.3	Normalization	19
4.3.4	Data to be supervised	19
4.3.5	The data frame is prepared	21
4.4	Glucose forecasting with numerical data	21
4.5	Glucose forecasting with wavelets images	24
4.5.1	Introduction to Digits	25
4.5.2	Creating wavelet images	25
4.5.3	Training with NVIDIA Digits	27
4.6	GlucNet Website	28
4.6.1	Connection with database	33
5	Experimental Results	34
5.1	Metrics	34
5.2	Results	35
5.2.1	First execution with Patient 1	35
5.2.2	Descriptions of the results	36
5.2.3	Train with Feed Forward	37
5.2.4	Train with LSTM	45
5.2.5	Train with NVIDIA Digits	53

6	Conclusions and Future work	55
6.1	General conclusions	55
6.2	Future work	56
7	Alejandro Varela work	58
8	Alvaro Delgado work	60
9	Acknowledgment	62
10	Bibliography	63

1. Summary

Abstract

Glucose levels prediction is a difficult task commonly faced by people with diabetes, a chronic health condition that affects how a human body synthesizes food. People with diabetes must have an exhaustive control of the levels of sugar in the bloodstream in order to manage insulin intakes, a procedure that is usually done manually and without enough accuracy. Levels of glucose in a human body depends on a lot of different factors, so the risk of doing miscalculations is always taken by the patient.

Nowadays, using new technologies such as Artificial Intelligence (AI) or Machine Learning (ML), these calculations can be supported and eased by the application of prediction systems. The field of AI and ML is very large, providing scientists with several different tools to create forecasting algorithms. During this project, we are going to focus on the creation and use of Neural Networks (NN) for glucose level prediction. To create this systems, we are exploring different types of Neural Networks (NN), ranging from regular numeric NN to graphic NN applications, which are vastly known in the world of data scientists.

However, these algorithms are always a difficult and hidden process for the real users, diabetes patients, so in order to make a higher impact on people affected by this disease, we will create an online application that will share all the power of NNs with the final patients, using a clear and intuitive user interface. In the meantime, patients will collaborate on the NN training process, as long as data provided by those using the application will allow us to develop a more heavily trained NN, thus improving its effectiveness in glucose levels forecasting.

Referring to the numerical NN and according to the results, we can state that their performance for 30 and 60 minutes prediction is quite accurate and, for 90 and 120 minutes prediction, it throws promising results.

Instead, for the graphical NN, even though the approach is interesting and the studies are showing us that the technology is very powerful, it needs much more investigation until we get good results.

Keywords

Diabetes Mellitus, Glucose Levels Prediction, Neural Networks, Forecasting, Online service, Machine Learning, NVIDIA Digits.

2. Introduction

2.1. Background

Diabetes is a chronic disease that is irreversible and affects the normal transformation of sugar in energy. It occurs when blood glucose, also called blood sugar, is too high in the bloodstream. Insulin, a hormone made by the pancreas, helps glucose from food get into the cells to be used for energy. Sometimes, the body doesn't make enough insulin or doesn't use insulin correctly, so glucose stays in the bloodstream and doesn't reach the cells. Over time, having too much glucose in the blood can cause several health problems. We can differentiate two main types of diabetes:

- **T1DM (type 1)**

This type is the less common within the patients of diabetes, a tenth of total. It is an autoimmune illness that attacks the pancreatic cells which generate insulin, making them dysfunctional. Thus, the patients need to supply themselves with daily doses of insulin that replace those the metabolic system should generate.

- **T2DM (type 2)**

With diabetes type 2, the metabolic system is able to generate insulin, but the immune system creates resistance to it, resulting in a similar outcome as in T1DM. Anyways, people with T2DM do not always need to supply themselves with daily doses of insulin. Normally, it is enough if they have a healthy routine and diet. This type of diabetes appears along the lifetime, and can be prevented or delayed with a healthy lifestyle (exercise, healthy diets...).

To maintain normal levels of glucose, the patients need to have control of those levels every time, taking into account the physical effort, diet and other factors such as alcohol intakes, level of stress, etc.

To measure the levels of glucose it is usual to have a manual glucose meter or a continuous glucose monitoring system. This is a tedious task that is done on a daily basis by the patient and that provokes an extra stress. Patients need to know the glucose level to decide if they need to take a corrective action, such as injecting insulin or eating some food. To help with this

routine, new methods are being investigated. One of them is the prediction of glucose levels, which could potentially allow patients to anticipate when there is a risk of hypoglycemia or hyperglycemia.

However, glucose levels prediction is a extremely difficult task because of the great amount of factors that are involved. Furthermore, predictions must be really accurate, since a wrong dose of insulin, taken from a wrong prediction, can cause health problems on the patient.

This field of study is not new. Previously, numerous techniques have been tried for glucose forecasting. The greatest aim of this techniques is to achieve a permanent solution for diabetes, such as the development of an "artificial pancreas". Figure 1 summarizes the different automatizing approaches for diabetes [1].

Method	By Hand	Semi-Automated	Artificial Pancreas
Bolus Estimation	Manual	Manual	Automated
Bolus Injection	Manual	Manual/CSIS	Automated
Basal Insulin Injection	Manual	Automated	Automated
Correction Calculations	Manual	Manual	Automated
Glucose Monitoring	CGMS/Manual	CGMS	Automated
CGMS calibration	Manual	Manual	Manual

Figure 1: Comparison of automatizing on current methods of diabetes treatment [1]

Through an artificial pancreas with an advanced prediction system along with measurement instrument such as a continuous glucose monitor, it could be possible to automate the insulin injection process in the same way as the pancreas of a healthy person works.

Some of the techniques used for this prediction model are: Genetic Programming, Random Forest Regression, K-Nearest Neighbors or Grammatical Evolution [1] and, of course, Neural Networks [2].

2.2. Goals

2.2.1. Development of glucose prediction models

The main goal of the project is to forecast the level of glucose that diabetes patients are going to have in the next 30, 60, 90 or 120 minutes based on their previous data. With this data, we'll train different versions of models to obtain as many results as possible. Thus, we can make comparisons between different models, and understand which parameters and configuration are the best for the glucose prediction. This will be a long-term study in the python environment to have a consistent model to train and predict glucose values.

2.2.2. Website service

Once we obtain a precise model with an optimal configuration of the data, we want to develop a website where any diabetes patient can access to make their glucose predictions with an intuitive interface. This application must be as simpler as possible to the user. With this platform, not only we provide the user with a prediction of its glucose levels, but also, if the user authorize it, we use his data to train our models. All the results and training will be saved in a database, that we will need to create and adapt to the current AbsysGroup database. Thus, there will be a connected system between our project and the previous one.

Hence, the main goal is to create the website system that will cover and be the final image of the models development and the machine learning work.

2.3. Work-plan

Our work-plan for the creation of GlucNet will be separated in the following stages:

- Study the information related to diabetes and the correlation between the different factors and the glucose level
- Get in touch with the data structures
- Pre-processing the data

- Start with the first neural network (NN) training
- Try different parameters and types of NN
- Implement the website
- Join the algorithm and the website
- Connect the database with the website

The rest of this document is organized as follows. Section 3 reviews previous approaches on glucose prediction by Neural Networks. Section 4 explains the whole process of the project including:

- Explanation the data format in sections 4.1 and 4.2.
- Data pre-processing task in section 4.3.
- Creation and training of numerical NN in section 4.4.
- Adaptation of the data for the graphical NN and its training in Nvidia Digits in section 4.5.
- Implementation of the website and its database in section 4.6.

Afterwards, section 5 explains the metrics used for the results and discuss the results from the different NN. Finally, on Section 6, we state our conclusions from the project and propose some future work to keep improving the project.

3. State of the art

There are several kind of Neural Networks and, due to the availability of new high performance computing architectures, the development and the applicability of them have increased exponentially during the last five years. The main types of NN are:

- LSTM
- RNN
- MLP
- CNN

LSTM networks have achieved very good performance in modeling several time-dependent phenomena. This is why most of the approximations found in the literature for the prediction of Blood glucose Levels are LSTM, since BG Data is usually obtained as a time series from CGMs.

Although most of the NN in the literature they are mostly devoted for time series predictions, there are also other Neural Network implementations following different schemes. In this section we made a revision of the recent solutions proposed for blood and interstitial glucose prediction using Neural Networks.

As with other neural network approaches, we can already find some solutions that tried to predict the level of blood glucose in the last years of the 20th century and early 21st century. Thus, perhaps the first approach that applied neural networks to predict the level of blood glucose in T1D was made in 1999 by Tresp et al. In [3] they train a recurrent neural network (RNN) that receives as inputs the levels of insulin, meals, exercise level inputs, and current and previous estimates of BG. The work is interesting but at the time it was developed obtaining this estimate was really difficult and therefore the data used for both training and testing are obtained from a single patient.

In the same line, Allam et al (2011) [4] proposed to use CGM signals to train an RNN to predict future glucose concentration values, considering

several prediction horizons. As in Tresp, the data used prevent the generalization of the model, to deal with data not used in training.

In [5] the authors make a glucose prediction using a sequential model consisting of a LSTM, a BI-LSTM, each with four units, and three full connected layers with 8, 64 and 8 units. They use both real and in silico patients. The data are pre-processed to solve two problems: outliers and lack of measurements. In order to generate a global model, a pre-training is performed with data from in-silico and real patients. Cross validation is used to avoid overfitting during the pretraining and training phase. To evaluate the results, the Root Mean square error (RMSE), Correlation Coefficient (CC), Minimum Time Lag and Fit metrics are used. Three methods have been used to make the prediction: Arima, SVR and LSTM, with prediction horizons of 15 minutes, 30 minutes, 45 minutes, 60 minutes. In the conclusions, the LSTM network surpassed the ARIMA and SVR methods. In comparison with these and in all the prediction horizons, the LSTM network reduced the RMSE and the TL, while the CC and the Fit were increased.

[6] develops a predictive model of BG to define future insulin therapy. The proposed architecture is called Deep Glucose Forecasting (DGF) and consists of two stacked LSTMs working in parallel. The first uses observed data and the second uses estimated data. The output of both branches is concatenated in a full connected layer. It performs justified data pre-processing because insulin and carbohydrate doses are 100 times higher than glucose measurements. In order to ensure that all values contribute equally to training these signals are re-scaled using their maximum and minimum values. It uses datasets obtained from UVA/PADOVA and real patients (PADOVA clinical center collected during pa@home Project experiments). Two different scenarios are used to simulate realistic intra-subject changes. To evaluate the results it uses the metrics Coefficient of Determination, Fittint Index, Root mean square error. The proposed solution is obtained as an average of the results obtained with 100 virtual adults. To improve the model, the weights of the LSTM are adjusted for a specific patient. The main limitation of the proposal is that the network is trained with measurements with noise, but the output has to be a signal without noise. Since the noise from the CGM measurements cannot be neglected, the network should try to reduce the noise from the output to improve the prediction. This is achieved by applying exponential filtering. The proposed architecture obtains a performance

similar to the current state of the art, both in in-silico patients, and in real patients, considering several prediction horizons 5, 10,...60.

In [7] a DNN with one LSTM layer and two full connected layers is proposed for BGL prediction using CGM data. The authors describe the method followed to set the architecture hyperparameters. Several experiments have been performed with the data of 10 patients to search for the model parameters. The performance is given in RSME. It compares with other LSTMs and with autoregressive models. The datasets used are obtained from Direct Net In-patient Accuracy Study provided by Diabetes Research in Children Network (direcNet). Data pre-processing is performed to eliminate redundancies and outliers between consecutive measurements. RSME is used to evaluate the model's performance. In addition, performance differences were assessed with the Wilcoxon test. The mean of the proposed model is 12.38 mg/dl while it is 28.84 and 50.69 for AR and LSTM.

Meijner and Persson (2017) [8] proposed to exploit LSTM in a model which takes CGM values, insulin dosages and carbohydrate intake as inputs. The proposal tackled only prediction horizons of 30 and 60 min. The training phase experimented initialization problems which lead to bad local optima.

Another interesting work is explained in [9]. This paper predicts glucose levels with a horizon of up to one hour. It uses a LSTM that only uses CGM values. The output of the model is the prediction plus the μ and σ^2 of the series, i.e. it indicates the certainty of the prediction which helps the users to interpret the result. It does not pre-process data except to multiply the glucose values by 0.01 to indicate that these values are in a range that can be predicted, therefore, when CGM values fail (there are gaps) it does not extrapolate values. He explains the method he has followed to select the hyperparameters of the LSTM network. It uses as dataset the Ohio T1DM Dataset for Blood Glucose Level Prediction of which it uses only two patients (Marling and Bunescu, 2018). Evaluates the prediction using the RMSE and Clarke error grid with irregular results.

Mirshekarian et al. [10] proposes an LSTM architecture for predicting blood glucose levels at 30 and 60 minutes. Perhaps the most interesting part of the paper is how they decide the architecture of the LSTM: they start from a physiological model and look for the dependencies between different

parameters of the physiological model. Then, they look for an LSTM architecture that maintains the dependencies of the physiological model. Their proposal indicates that the obtained architecture can be easily extended to other parameters. The architecture contains three layers, the input one, the hidden one, which is an LSTM, and the full connected output one. This last one has two additional nodes, one for the prediction at 30 minutes and the other at 60 minutes. They have assembled a database containing approximately 1600 days of actual patient information. This data includes insulin, food, exercise and sleep. They perform a pre-processing: the BG is scaled by 0.01, the rest of the values are normalized to [0.1]. Lost BG values are interpolated linearly, however, they are not used as predictive targets during training. Compare the results with those obtained with the Support Vector Regression (SVR) indicating that they are competitive. They also claim that the best thing about the model is that it can incorporate new physiological parameters without the need for sophisticated manual engineering.

Also by Mirshekarian’s group [11] proposes a novel architecture called memLSTM which is a mixture of LSTM with a type of networks called neuronal attention models or augmented memory models. The idea behind the augmented memory networks is to give the model access to an external memory that can be written during the training and test phases. The architecture is divided into three modules: LSTM, memory module and feedforward module. The authors use two datasets. One of them is the real OhioT1DM that contains data from 6 patients, with more accurate BGL measurements and more reliable information about food and insulin. More accurate Medtronic EnliteR sensors are used along with additional physiological data such as heart rate (HR), skin temperature (ST), and skin conductance (SC) provided by the Basis Peak fitness band and, finally, carbohydrates introduced via the Bolus Wizard. This data set is available to the scientific community and is part of the used in this work. On the other hand, they use insilico patients from two simulators: AIDA and UVA/PADOVA.

The objective of the work in [12] is the application of machine learning to the prediction of glycemia levels that can be used to generate alerts. In the article the authors try to demonstrate that the metric should be particularized for the range in which the highest accuracy is desired. Usually the accuracy is computed over the total glycemic range which can lead to erroneous models. One solution to this problem is to break the glycemic

range into significant parts and analyze each subrange independently. The article also tries to demonstrate that a pre-processing of the data is necessary to take into account the imbalances that appear in the data, since most of the time the patients are in a region where the glycemia is normal. An adaptation of standard pre-processing techniques has been made for imbalanced labelled data. The latter increased accuracy in particular glycemic range subranges. In the experiments it uses as dataset the last six patients incorporated in the Ohio T1D dataset. In total, the work studies the behavior of 10 regression models among which the Multilayer Perceptron and four oversampling techniques to deal with data imbalances stand out: Random Oversampling, Synthetic Minority Oversampling Technique, Adaptive Synthetic Oversampling Technique. It uses two metrics to evaluate the performance of Mean Absolute Relative Difference (MARD) prediction and Clarke Error Grid Analysis. The MARD specifies it for four different cases: all regions, regions with hypoglycemia, regions with normoglycemia, and with hyperglycemia. To carry out the study, the 10 machine learning techniques proposed are combined with the 4 oversampling options, giving rise to 40 different combinations. The results show that if a hypoglycemic alert is desired, a trained MLP in conjunction with an oversampling method produces the highest accuracy. Therefore, it can be deduced that the choice of the best algorithm depends on the glycemic range.

Convolutional RNNs have also been explored to predict the BG level. Although more used in image processing CNN can be adapted for this problem. For instance (Li et al., 2018) [13] transformed time series of glucose level, carbohydrate and insulin, by concatenating them and preprocessing using deep convolutional networks: Then the recurrent LSTM layers accepts these features instead of the CGM measurements directly. The model is trained on 10 in silico adults patients from the UVA/Padova simulator.

Other works:

- Prediction of Blood Glucose Levels And Nocturnal Hypoglycemia Using Physiological Models and Artificial Neural Networks [14]
- Predicting hypoglycemia in diabetic patients using time-sensitive artificial neural networks [15]
- Hypoglycemia prediction using extreme learning machine (ELM) and

regularized ELM [16]

- Evolvable rough-block-based neural network and its biomedical application to hypoglycemia detection system [17]
- Artificial neural network algorithm for online glucose prediction from continuous glucose monitoring [2]

4. GlucNet

GlucNet is an online service for glucose classification and prediction that is based in Neural Networks implemented in Python.

The whole process to get the final product includes:

- Working with the data and preparing it to every different scenario.
- Creating the neural nets that will work in the website.
- Exploring a new approach on glucose prediction and classification based in Graphic NN.
- Creating a website where NN will be supported.

4.1. Devices

The following sensors are used to measure the glucose and insulin values, physical activity and food intakes:

- Free Style Libre-Abbott
- Minimed Medtronic CGM
- Medtronic Insulin Pump
- Roche Insulin Pump
- Fitbit Ionic
- Notes from patients

Please note, that not all the patients have all the data because we were not able to recover all the items. Using this devices, we recorded the following data:

- Interstitial glucose
- Notes of estimated carbohydrate units ingested, taken by each patient.

- Insulin injected using an insulin infuser device from Medtronic and/or Roche, which registers injections of both basal and bolus insulin every five minutes.
- Burned calories and steps
- Other information

4.2. Data sources

In this section we will show the two different sources of data we got to work with.

Our main source of data and the one that is going to guide the study are the files containing data from Patients of the Hospital Universitario Príncipe de Asturias de Alcalá de Henares in Madrid, from June 13th, 2018 to, July 17th, 2019.

Column	Meaning	Measuring Device	Note
Basal Rate (U/h)	Basal rate	Medtronic Pump	Every 5 min if it is ok
Bolus Type	Bolus type	Medtronic Pump	anytime,
Bolus Volume Delivered (U)	Bolus Units	Medtronic Pump	anytime,
BWZ Carb Ratio (U/Ex)	Insulin-to-carbohydrate Ratio	Medtronic Pump	anytime,
BWZ Carb Input (exchanges)	Amount of carbohydrates	Medtronic Pump	anytime,
Sensor Glucose (mg/dL)	The sensor glucose value	CGM Medtronic	Every 5 min if it is ok
Hora	Time	Anyone	
Calorias	Calories from Fitbit	Fitbit	anytime,
Ritmo Cardíaco	HR from Fitbit	Fitbit	anytime,
Estado	Sleeping state from Fitbit	Fitbit	anytime,
Pasos	Steps from fitbit	Fitbit	anytime,
Histórico glucosa (mg/dL)	Automatic glucose level	Freestyle	Every 15 min if it is ok
Glucosa leída (mg/dL)	Manual glucose level from FreeStyleLibre	Freestyle	anytime,
Insulina de acción rápida sin valor numérico	Bolus Mark	Freestyle	anytime,
Insulina de acción rápida (unidades)	Bolus units	Freestyle	anytime,
Alimentos sin valor numérico	Food intake	Freestyle	anytime,
Carbohidratos (raciones)	Food intake units	Freestyle	anytime,
Insulina de acción lenta sin valor numérico	Slow action insulin timestamp	Freestyle	anytime,
Insulina de acción lenta (unidades)	Units of slow action insulin	Freestyle	anytime,
Notas	Notes	Freestyle	anytime,
Glucosa de la tira (mg/dL)	Glucose Meter from FreeStyle Libre	Freestyle	anytime,
Cetonas (mmol/L)	Ketones from Freestyle Libre	Freestyle	anytime,
Insulina comida (unidades)	Insulin units	Freestyle	anytime,
Insulina corrección (unidades)	Units from corrector bolus	Freestyle	anytime,
Insulina cambio usuario (unidades)	Daily insulin units	Freestyle	Low fidelity
Hora anterior	Time change	Freestyle	anytime,
Hora actualizada	Time change	Freestyle	anytime,
Sensor Calibration BG (mg/dL)	Glucose calibration	Extern glucometer	anytime,
BG Reading (mg/dL)	Glucose calibration	Extern glucometer	anytime

Table 1: Meaning of each parameter from Alcalá de Henares patients datasets

Table 1 shows the variables stored on the csv file with their meanings, the devices from where data was retrieved, an explanation and notes on the frequency of acquisition and/or any other particularity. We have access to 15 data-sets of different patients from this hospital, each one with around 3500 rows of data along with the timestamps.

Our second source of data are the files from 6 different patients from Ohio's Hospital. These csv have very different columns and almost all of them do not have enough data to be useful (less than 25% of the total amount of rows in the data-set). The columns that will have more importance for us are "basaleventvalue", "accelerationeventvalue" and "Glucose_levelvalue". From the first two we can get the equivalent Basal Rate (U).

4.3. Data preprocessing

As in every data-mining and machine learning project, the first task after analysing the data is preparing it to be used by the algorithm. In this glucose prediction project we are working on data from single individuals based on time series.

4.3.1. Study of the data-set

Firstly, when we take a look at the main source of data (data from Patients of the Hospital Universitario Príncipe de Asturias de Alcalá), we noticed that the majority of the columns do not provide useful information as they have repeated information or not to many values to work with. This columns that are not useful or have too much "not assigned" (NaN) have to be deleted. Furthermore, as we are working with data based on time, more specifically every 5 minutes, it is important to treat the data as a time series. The column of time is not only a variable, but also the value that specifies the order in which the data is related. To specify this order we use the time as an index, making it easier to train the data. In the case of the Basal Rate, it is valued as units per hour so it is necessary to divide every value in this column by 12, resulting in units per 5 minutes. Figure 2 shows the result of dropping the columns from the data-set.

Hora	Histórico glucosa (mg/dL)	Glucosa leída (mg/dL)	Calorías	Ritmo Cardíaco	Estado	Pasos	Basal Rate (U/h)	Bolus Type	Bolus Volume Delivered (U)	BWZ Carb Ratio (U/Ex)	BWZ Carb Input (exchanges)	Sensor Glucose (mg/dL)
2018-06-13 17:20:00	488.0	NaN	4.0250	NaN	NaN	0.0	1.5	NaN	NaN	NaN	NaN	NaN
2018-06-13 17:25:00	NaN	NaN	4.0250	NaN	NaN	0.0	1.5	NaN	NaN	NaN	NaN	NaN
2018-06-13 17:30:00	NaN	NaN	4.0250	NaN	NaN	0.0	1.5	NaN	NaN	NaN	NaN	NaN
2018-06-13 17:35:00	469.0	NaN	4.0250	NaN	NaN	0.0	1.5	NaN	NaN	NaN	NaN	NaN

Figure 2: Resulting data-set after dropping columns

Furthermore, there is another data source, the Ohio’s Hospital, which has very different columns compared to our previous source. As we want to create an unified model that can support different data sources, we need to adapt this data-set to our main structure.

As we did with the other data-sets from Principe de Asturias Hospital, we need to drop all the columns that do not have sufficient information. In this case, the only columns that have good and enough data to train are the levels of glucose and the Basal Rate. This data-set is also organized in 5 minutes timestamps.

4.3.2. Data filling

Once the data frame has the proper columns and is converted into a time series, the next step is to create data for the non assigned values, this task is commonly known as data-filling. In this case, it was necessary to use two different types of data-filling, interpolated and bfill. The interpolate function fills the values between 2 different numbers making use of a lineal function, and the bfill function simply copies the data from the last seen value in all subsequent missing values.

We use the interpolate function to fill the intermediate gaps in the numerical columns such as the levels of glucose to obtain reasonable new values. Using interpolate for the basal rate won’t be accurate since the values are taken every hour. in this case, we simply copy the value for 5 minutes in every hour. Finally, for the steps and calories columns it is not necessary to

perform any data-filling.

4.3.3. Normalization

On the machine learning work, every type of column usually have very different ranges of values which are not homogeneous. Normalization is the best solution to have similar ranges to work with. It is sometimes required to normalize the data for training with some neural networks such as Keras. In the case of the numeric neural nets, we use the MinMax scaler between 0 and 1 because the data-set does not contain negative values.

Instead, when using graphic neural networks, we apply the Zscore normalization to represent the data graphically and train the algorithm in Digits. We apply this different method for the graphical data because with MinMaxScaler the graphics for the columns with low values did not show the range properly. Zscore applies one normalization for each type of data.

4.3.4. Data to be supervised

Once we have all the columns with the right data, we need to convert the data-set in a supervised data-set. This is a required task because we need all the data related to each prediction in each line. More deeply, for each row, which target is 30, 60, 90 or 120 minutes prediction, we need the data that affects directly to the prediction.

As we see in the Figure 3, it is always necessary to have the values of glucose and basal rate from the previous 120 minutes and the data of the basal rate from 30 to 120 minutes in the future.

To do this, we copy the glucose and basal rate data of the previous 24 rows and the data of the future 6-24 rows in each row. In Figure 4 the function that performs it is shown.

The function that converts the data frame to supervised data, using the function on Figure 4, depends on the type of prediction you want to perform, having as parameter the minutes for the prediction.

Variable	BNF T	Available? (⊗)			
		30	60	90	120
y_r	Target				
Time	x1	⊗	⊗	⊗	⊗
$G_{(t)}$	x2	⊗	⊗	⊗	⊗
$G_{(t-[120...90])}$	x3	⊗	⊗	⊗	⊗
$G_{(t-[90...60])}$	x4	⊗	⊗	⊗	⊗
$G_{(t-[60...45])}$	x5	⊗	⊗	⊗	⊗
$G_{(t-[45...30])}$	x6	⊗	⊗	⊗	⊗
$G_{(t-[30...15])}$	x7	⊗	⊗	⊗	⊗
$G_{(t-[15...0])}$	x8	⊗	⊗	⊗	⊗
$I_{(t)}$	x24	⊗	⊗	⊗	⊗
$I_{(t-[120...90])}$	x25	⊗	⊗	⊗	⊗
$I_{(t-[90...60])}$	x26	⊗	⊗	⊗	⊗
$I_{(t-[60...45])}$	x27	⊗	⊗	⊗	⊗
$I_{(t-[45...30])}$	x28	⊗	⊗	⊗	⊗
$I_{(t-[30...15])}$	x29	⊗	⊗	⊗	⊗
$I_{(t-[15...0])}$	x30	⊗	⊗	⊗	⊗
$I_{(t+[0...15])}$	x31	⊗	⊗	⊗	⊗
$I_{(t+[15...30])}$	x32	⊗	⊗	⊗	⊗
$I_{(t+[30...45])}$	x33	✗	⊗	⊗	⊗
$I_{(t+[45...60])}$	x34	✗	⊗	⊗	⊗
$I_{(t+[60...75])}$	x35	✗	✗	⊗	⊗
$I_{(t+[75...90])}$	x36	✗	✗	⊗	⊗
$I_{(t+[90...105])}$	x37	✗	✗	✗	⊗
$I_{(t+[105...120])}$	x38	✗	✗	✗	⊗

For this purpose, all the historical data are presented as variables as well as the future values of insulin and meals valid for each prediction horizon. The table shows all the variables with their corresponding historical or future data. For each horizon an (⊗) indicates that the variable is available for prediction, while a (✗) indicates that it is not permitted

Figure 3: Explanation of variables needed for supervised learning

```
def series_supervised(data, label, n_in = 1, n_out = 1, dropnan = True):
    n_vars = 1 if type(data) is list else data.shape[1]
    df = pd.DataFrame(data)
    cols, names = list(), list()

    for i in range(n_in, 0, -1):
        cols.append(df.shift(i))
        names += [('%(t-%d)' % (label, i*5)) for j in range(n_vars)]

    for i in range(0, n_out):
        cols.append(df.shift(-i))
        if i == 0:
            names += [('%(t)' % (label)) for j in range(n_vars)]
        else:
            names += [('%(t+%d)' % (label, i*5)) for j in range(n_vars)]

    agg = pd.concat(cols, axis=1)
    agg.columns = names
    # drop rows with NaN values
    if dropnan:
        agg.dropna(inplace=True)
    return agg
```

Figure 4: Function that prepares the data for supervised learning

Taking this argument, the function calculates how many columns are needed to create the supervised data-set: always 24 columns for the past 120 minutes of glucose and 24 columns for the last 120 minutes of insulin intake. It also creates columns for the future insulin intakes; this is useful because it is one of the main reasons why glucose level can fall. It is also needed to crop the data-set since, at the beginning, we do not have past values for insulin, and at the end, we do not have future values.

4.3.5. The data frame is prepared

After this, the data frame is already prepared to work with neural networks, but some other tasks could be done to improve the performance of the algorithm. One of them is the Data augmentation, this means creating totally new data based on given data frames. In the Medicine-AI field this can be really useful since one of the main problems is the lack of data. Creating new data makes possible to work with more examples, thus creating a more accurate neural net.

4.4. Glucose forecasting with numerical data

In this project we are dealing with a multi-variable regression problem, as the goal is to predict the glucose that a diabetes patient can have in a determined time. Through the data conversion to time series, we can transform the data to have a supervised data-set. This means that the algorithm is trained with the data of the patient before the consultation of the level of glucose, and with the data of the patient after that consultation (figure 5).

Due to some patients have less data parameters than others, we had to divide them and create different models depending on the parameters they have. Thanks to this, we can evaluate the training performance depending on the type of data, and get some conclusions.

```
Index(['G(t-120)', 'G(t-115)', 'G(t-110)', 'G(t-105)', 'G(t-100)', 'G(t-95)',
      'G(t-90)', 'G(t-85)', 'G(t-80)', 'G(t-75)', 'G(t-70)', 'G(t-65)',
      'G(t-60)', 'G(t-55)', 'G(t-50)', 'G(t-45)', 'G(t-40)', 'G(t-35)',
      'G(t-30)', 'G(t-25)', 'G(t-20)', 'G(t-15)', 'G(t-10)', 'G(t-5)', 'G(t)',
      'G(t+5)', 'G(t+10)', 'G(t+15)', 'G(t+20)', 'G(t+25)', 'G(t+30)',
      'I(t-120)', 'I(t-115)', 'I(t-110)', 'I(t-105)', 'I(t-100)', 'I(t-95)',
      'I(t-90)', 'I(t-85)', 'I(t-80)', 'I(t-75)', 'I(t-70)', 'I(t-65)',
      'I(t-60)', 'I(t-55)', 'I(t-50)', 'I(t-45)', 'I(t-40)', 'I(t-35)',
      'I(t-30)', 'I(t-25)', 'I(t-20)', 'I(t-15)', 'I(t-10)', 'I(t-5)', 'I(t)',
      'I(t+5)', 'I(t+10)', 'I(t+15)', 'I(t+20)', 'I(t+25)', 'I(t+30)',
      'Glucosa leida (mg/dL)', 'Calorias', 'Ritmo Cardiaco', 'Pasos',
      'BG Reading (mg/dL)', 'Bolus Volume Delivered (U)',
      'BWZ Carb Ratio (U/Ex)', 'BWZ Carb Input (exchanges)',
      'Sensor Calibration BG (mg/dL)', 'Sensor Glucose (mg/dL)'],
      dtype='object')
```

Figure 5: Columns of the supervised data-set

Hence, these are the types of model depending on the parameters of each patient:

- Patients with insulin and basic values (3 patients)
 - Glucose
 - Steps
 - Heart rate
 - Calories
 - Basal rate
- Patients with only basic values (10 patients; the patients with insulin values are also valid for this training)
 - Glucose
 - Steps
 - Heart rate
 - Calories
- Patients from Ohio and 3 from Alcalá de Henares that contains:
 - Glucose
 - Basal rate
- All the patients, but only with the glucose values.
 - Glucose

The initial model we chose to train the data was a simple feed forward declared with Keras to test if the supervised data-set generated was well configured.

Once checked that the data pre-processing was properly prepared, we configured the first type of neural network to train and test all the patients. The function that creates this NN can be seen on Figure 6.

```
def crear_modeloFF(number_columns):  
    model = Sequential()  
    model.add(Dense(number_columns, input_shape=(1,number_columns),activation='tanh'))  
    model.add(Flatten())  
    model.add(Dense(1, activation='tanh'))  
    model.compile(loss='mean_absolute_error',optimizer='Adam',metrics=['mse','accuracy'])  
    model.summary()  
    return model
```

Figure 6: Creation of the FF model

This initial neural network is configured with the input layer (data supervised), one hidden dense layer with a hyperbolic tangent activation function that expects rows of data with the number of columns of the supervised data-set, and one output dense layer with one node.

- 190 epochs
- Tahn activation function
- Adam optimizer
- Input shape: (1, number of columns of the data-set)
- Mean absolute error as loss function

After testing the models mentioned with the feed forward neural net, the next step was to modify the configuration of the model to see if it was possible to improve the results.

There are different variations of neural networks according to the type of data. Due to data-sets are time series, we decided to test configurations with a type of recurrent neural network called LSTM (Long short-term memory).

These types of neural networks were created to give a solution to the short-term memory. They have an internal mechanism called gates that can regulate the flow of information. These gates can learn which data in a sequence is important to keep or throw away. By doing that, it can pass relevant information down the long chain of sequences to make predictions.

This is the configuration of the new neural net:

```
model = Sequential()  
model.add(LSTM(number_columns, input_shape=(1,number_columns),  
activation='relu', return_sequences = True))  
model.add(Flatten())  
model.add(Dense(1, activation='tanh'))  
model.compile(loss='mean_absolute_error', optimizer='Adam',  
metrics=["mse", 'accuracy'])
```

4.5. Glucose forecasting with wavelets images

For this different approach of glucose forecasting, we face a classification problem: we are going to predict whether the patient is going to have a hypoglycemia episode in the next period based on images with data from 6 to 24 hours.

These images are divided in two classes. Taking a period from 6h to 24h (depends on the model), we will seek in the following period for values of glucose below 70 mg/dL. If that period has at least one value below 70 mg/dL, it will be considered as a Hypoglycemic period, otherwise, it will be considered as a Normal period.

This type of training with images is something totally new in the field of diabetes-AI, so it will be an experimental work.

To prepare the data for this section some pre-processing tasks are avoided such as making the data set supervised.

To support the neural nets and its training we are going to use Nvidia Digits, a platform that eases the work with images.

4.5.1. Introduction to Digits

The NVIDIA Deep Learning GPU Training System (DIGITS) puts the power of deep learning into the hands of engineers and data scientists. DIGITS can be used to rapidly train the highly accurate deep neural network (DNNs) for image classification, segmentation and object detection tasks.

DIGITS simplifies common deep learning tasks such as managing data, designing and training neural networks on multi-GPU systems, monitoring performance in real time with advanced visualizations, and selecting the best performing model from the results browser for deployment. DIGITS is completely interactive so that data scientists can focus on designing and training networks rather than programming and debugging.

4.5.2. Creating wavelet images

As we explained in subsection 4.5, NVIDIA Digits works with images. Until now, all our data was based in a numerical data-set. Then, we have to transform all our data to feed the graphical NN.

Firstly, we tried to create linear representations of the data with all the columns we had (Glucose levels, Basal Rate, Heart Rate, Calories and Steps), resulting in graphics such as the one in the Figure 7.

After training and seeing the results with this type of representation, we could agree that the neural net was not able to find any pattern on those images. We tried to apply different normalization to the parameters to give equal visualization to each of them, but it resulted on the same outcome.

The problem with this type of representation is that almost the whole image does not have information (white background) and the changes on the values of the different columns can not be appreciated correctly.

After that and taking into account our previous work, we decided to use only the glucose levels transformed in a wavelet image.

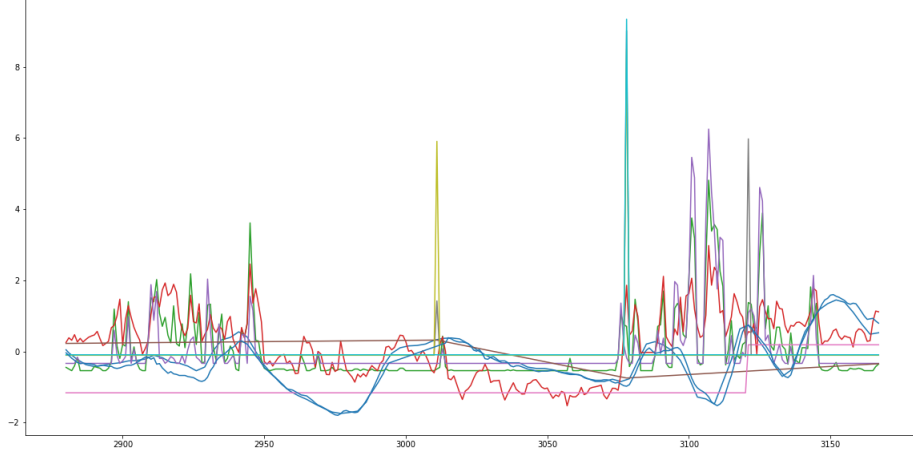


Figure 7: Linear representation of the data-set in a period of 24 hours

This wavelet images are the result of plotting the wavelet power spectrum of a single time series. The result from this technique are images similar to heat-maps that concentrate all the information from a period of time.

The vertical axis shows the Fourier periods and the horizontal axis shows time step counts. The color levels can be defined according to quantiles of values or to equidistant breakpoints (covering the interval from 0 to maximum level), with the number of levels as a further parameter.

In our case, we tried to create images with different configurations to find the one that was more informative. Regarding to the last linear image, which had a lot of useless information such as the background and the axis (axis can be useful for human understanding but NN do not need them), we decided to delete as much of them as we could. Therefore, we created images with the next configuration both for equidistant intervals, shown in Figure 8, and quantile intervals, shown in Figure 9:

- 500 levels
- Plot contour and ridge
- Plot cone of influence

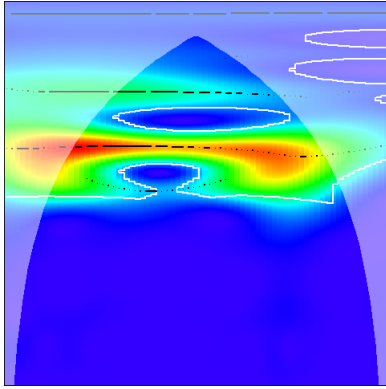


Figure 8: Representation of glucose levels during 6h with equidistant intervals

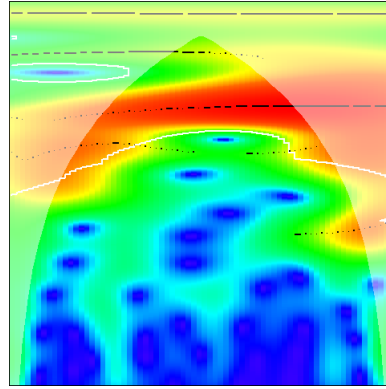


Figure 9: Representation of glucose levels during 6h with quantile intervals

Finally, these are the images that we are going to use for the training phase in NVIDIA Digits, both for 6h and 24h periods.

4.5.3. Training with NVIDIA Digits

Once we have all our data transformed into images, training comes into action. Training with NVIDIA Digits give us a lot of possibilities, since creating and training a NN from scratch to using a downloaded pre-trained model (model that has been extensively trained with other data-set and has already its own weights) from the catalogue of Digits to train with. In our case, we will use the pre-trained models that come with Digits: AlexNet, LeNet and GoogleNet. This will allow us to get results faster than starting from scratch, which can be really hard if you do not have enough data or knowledge about graphic NN.

After the first trials, we decided that the AlexNet pre-trained model was the one that fitted better for our problem. Hence, we tried different configurations of the following parameters:

- Training epoch
- Base learning rate
- Solver type

- Other options about learning rate

After deeply analysing the problem, we decided to focus on the training with 6h periods since 24 hours was not a period precise enough to be informative about hypoglycemic episodes. We extracted a total of 1521 images from the data (381 from Hypo class and 1140 from Normal class).

4.6. *GlucNet Website*

After we decided the type of tool we wanted to use (Python numerical models instead of NVIDIA Digits, not only because we obtained better results with the former but also because the latter was still in development), the next step was to build a visual and intuitive platform for the patients both to have the possibility to predict their values of glucose and to save all the information in a database.

For this purpose, the first step was to choose a framework able to execute python scripts. Since we had no experience on web developing at the start of the project, we decided to use Django framework. Django is a high-level Python Web framework that encourages rapid development, clean and pragmatic design. Moreover, Django fits our project perfectly given that all our models and scripts are implemented in Python.

For the front-end of the website, we will use a template to make the web design phase easier and faster. The template we used is '11 Ozberk' from Colorlib, which allowed us to create a clean and material design interface, easy to navigate and user-friendly for patients.

The website is compounded of two static elements (footer and navigation bar) that appear in every screen and the dynamic container, which shows the information for each different screen.

In order to connect the template to the back-end part, we are using Django. Django framework provides different python classes to fill, where you specify the schema of the back-end. Hence, the task of connecting our python scripts is really intuitive, due to the fact that the environment is full Python. The class `views.py` contains different functions for each screen of

the web, where the data obtained from the template is collected and used in these functions to execute our scripts. Then, based on the request method, we pass the information from our scripts to the HTML template.

The final website consists on the following screens:

- Landing Page: It is a simple main page with information about the purpose of the website and the people involved in the project. Its objective is to be a good starting point to get in touch with the website, with not too many elements that could overwhelm the user experience.
- Log in page: Standard page that users will use to log into the website. The benefits from being logged in the website are:
 - Train the models with your own data. This will help the model to understand different types of patients and give more accurate predictions.
 - Create personal models that are only used and trained with the data they upload. Sometimes, the personal models can be more accurate since it will get the characteristics of a unique patient.
 - Save the results of user's training so they can look at them later.
 - Create a single prediction based only on the 25 values of glucose from the last 120 minutes.
 - Collaborate on the creation of future models with their data.
- Sign up page: Standard registration page where a new user is created. The fields required for it are: name, surname, email and password. Also, the user has to give permission for the usage of their data uploaded.
- Single prediction page: A page where the user can make a single prediction without training. This prediction is done through the glucose only model with the data of the last 120 minutes of glucose. There is an explanation in the upper part about the required format of data (25 values from -120 minutes to 0 minutes one every 5 minutes, separated by ";"). A caption of the page's body can be seen in Figure 10.

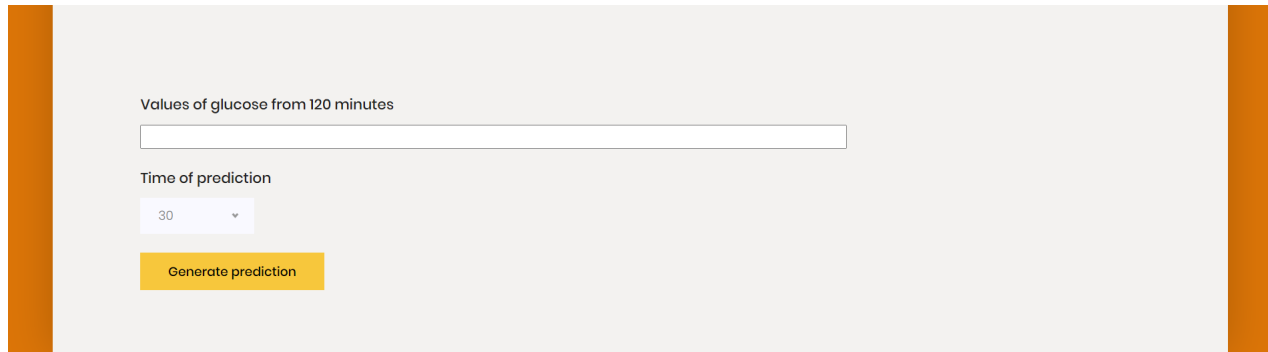
The image shows a web interface for a single prediction. It has a light gray background with orange vertical bars on the left and right. At the top, there is a label "Values of glucose from 120 minutes" above a long, empty white input field. Below this, there is a label "Time of prediction" above a dropdown menu showing "30" with a small downward arrow. At the bottom, there is a yellow button with the text "Generate prediction".

Figure 10: Single prediction page

- Training page: Training screen, shown in Figure 11, where the user can upload the data to see the performance of the NN. If the user is registered, the NN will also be trained with the data. To select the model, some fields have to be filled before the training:
 - Type of model: Personal or General (Only general if the user is not logged)
 - Time of prediction (30, 60, 90 or 120)
 - The model to be used based on the columns of user's data.
 - In the case of personal models, selection of the specific model from the last 5 personal models.

When working with automated services, it can be really difficult to handle different types of data.

To make the process easier, we determine the shape of the data the user uploads. The requirements for the uploaded .csv to ensure an optimal performance of the model are:

- The columns need to have the following names "Glucosa" for the Glucose levels, "Pasos" for the steps, "Ritmo Cardiac" for the heart rate and "Basal Rate (U/h)" for the insulin supplied by the pump. If there are extra columns in the .csv, no problem arises, only the columns of the model will be used.
- The .csv needs to have one row each 5 minutes.

Upload your data and configure the model

Upload your personal data and get the predictions
The prediction returns a clarke graphic that represents the accuracy of the model, a lineal representation of the glucose and the RSME.

Personal or General Model?

There are 2 types of model, General and Personal (the Personal model is only available if the user is registered). The general model is trained with the data from every user that use it. The personal model is created and trained personally for each user. Take into account that the more data the model gets, the more accurate it is, so if you do not have a lot of data the general model can give better results. Anyways we invite you to test both of them and see the difference.

General

Type of model

1. Model with only glucose values ☒

2. Model with glucose, calories, heart rate, steps and insulin values ☐

3. Model with glucose, calories, heart rate and steps values ☐

4. Model with glucose and insulin ☐

Time of prediction

30

Version of model (most recent first)

Last trained model

Seleccionar archivo Ningún archivo seleccionado

Upload data and generate predictions

Figure 11: Training page

- If the .csv does not have values of glucose for every 5 minutes, the missing fields need to be empty in order to recognize it as NaN.
- Numbers need to use decimal dot.
- **User page:** On the user screen, shown in Figure 12, the user personal information is displayed. This includes: name, surname, email and username. Also, we display a small summary of the last 5 training results that the user has performed.
- **Results page:** This page is the one that comes after training is performed. It shows the results of the training and prediction with the data uploaded. The information displayed, as seen in Figure 13, is the RMSE (Root Mean Squared Error), the Clarke error grid along with the number of points on each section and a graph that compares the real target with our prediction. We also make a single prediction for the last 120 minutes in the data-set, simulating how the algorithm would work in real life usage.

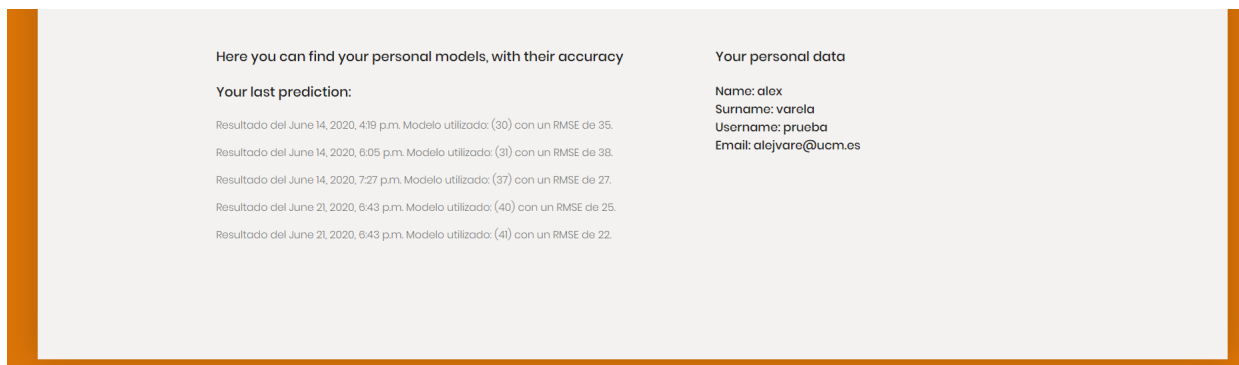


Figure 12: User page

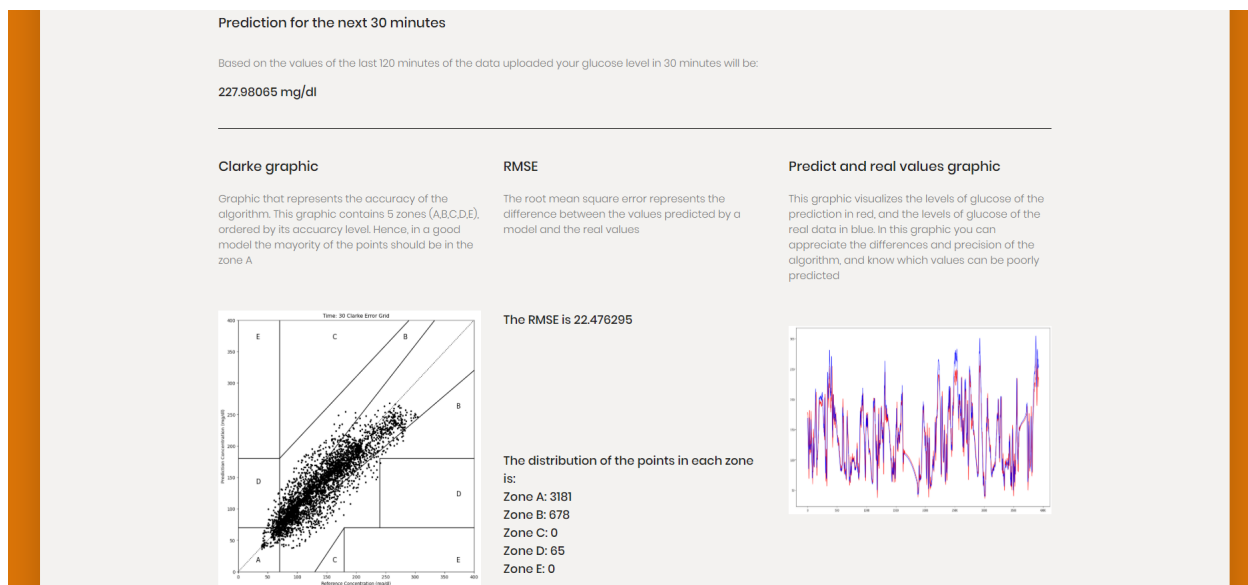


Figure 13: Results page

4.6.1. Connection with database

Given the fact that, in this website, the user creates new models, uploads data and the system generates results, it was necessary to implement a database to keep stored all this information. Furthermore, due to there was an implemented database from other final degree project that was functioning, we needed to follow a established syntax to create a database which was adaptable to the existing one.

The needed tables to incorporate to the system are the following:

- User tables that Django creates automatically. We are using the table 'auth-user' for users management.
- Files tables where the system keeps the uploaded files of the user, with the user id.
- Models table that saves the type of model, the last user that trained it, whether it is personal or general and the model RMSE.
- Results table to keep the results of a prediction (RMSE, zones of the clarke graphic, user id).

To create the database, we used MySQL Workbench and Server, which are really intuitive to manage the different components of the data structure.

5. Experimental Results

5.1. Metrics

In order to measure the efficiency and behaviour of the model, we are going to use the following metrics:

- Clarke error grid

This Clarke graphic visualizes black points that represent the real test target (axis x) and the predicted values (axis y). In a first instance, the closer the points are allocated around the diagonal (which means that predictions are equals to real test set), the more precise the model will be.

This Clarke graphic displays different zones: A,B,C,D,E. These zones represent the validity of the precision. So the target of the model is to have most of the point within the zones A and B, and as few points as possible in the zones C, D and E. An example of an empty Clarke Error Grid can be seen in Figure 14.

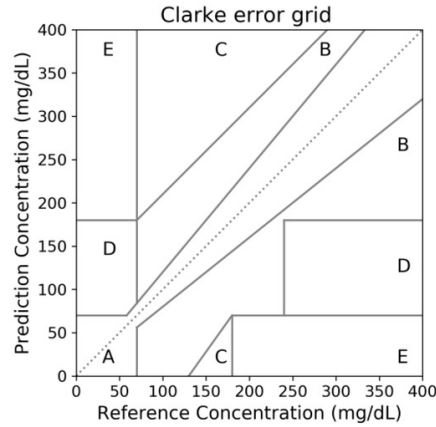


Figure 14: Clarke Error Grid

- RMSE

To have more measurements apart from the Clarke image and the number of points in each zone, we calculate the RMSE i.e root mean square

error, following the formula in Figure 15. Due to the RMSE is calculated over glucose values that usually range between 40 and 500 mg/dl, a good RMSE should not exceed 30 mg/dl.

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (x_{1,i} - x_{2,i})^2}{n}}$$

Figure 15: Root Mean Squared Error formula

5.2. Results

In this section we are going to document all the results obtained after testing different techniques to predict glucose values.

5.2.1. First execution with Patient 1

Our first contact with a real prediction was training the Patient 1 from the Hospital Universitario Príncipe de Asturias. The patient data-set contained 3824 rows of data after pre-processing, enough to split it into a train and test sets to try out a first execution.

The first configuration to generate the supervised data with time series (section 4.3.4) was with 30 minutes, i.e, the train data contains all the information of the patient 120 minutes before the time 0, and the target are all the glucose values after 30 minutes.

We used the first configuration of the model (Feed forward described in section 4.3) to test a first execution. The results can be seen in Figure 16.

In this first prediction over the test set from the Patient 1, most of the points in the Clarke error grid fall in zone A, but some points are in the critical areas.

However, the conclusion of this first execution is that, considering this is a simple model and we only trained 70% of the data-set of one patient, the results are promising with this type of supervised data, with the time series

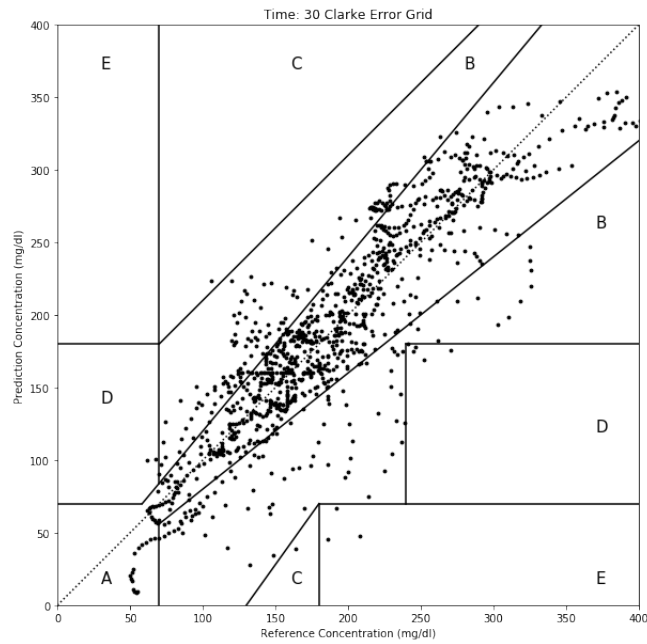


Figure 16: Results of the first training with the supervised data-set

and with the parameters trained.

This execution is documented in the file 'Hello diabetes.pynb'

5.2.2. Descriptions of the results

To study the different results, we use the following dynamic:

1. Create the model adapted to the type of data

```
#Trained with patients that contains insulin values
model_30_insulin = crear_modeloFF(number_columns)
```

2. Train the model for each patient and save the model

```
execute(30, pat_7_insulin, model_30_insulin, True)
execute(30, pat_8_insulin, model_30_insulin, True)
execute(30, pat_9_insulin, model_30_insulin, True)
model_30_insulin.save('Models/model_30_insulin.h5')
```

3. Create a new model for each patient and compare the results of the new model with those of the pre-trained model

```
print(" Patient 7 No training")
execute(30,pat_7_insulin , crear_modeloFF(61) ,True)
print(" Patient 7 Trained")
execute(30,pat_7_insulin , trained_modeldl_30_insulin ,False)
```

5.2.3. Train with Feed Forward

The next step was to try this basic model with more patients, and study if training a single model with more than one different patient returns better results. Therefore, we needed to automate the process of preparing the patients (fill NaN values, cut the data-set only with the rows with data, drop useless columns...).

All these tests and studies are collected in the file 'Model Dense layer (original FF).pynb'. We are using all the patients with data from Hospital Universitario Príncipe de Asturias and from Ohio's Hospital.

These first detailed results are for prediction of glucose for 30 minutes:

1. Model with insulin and basic values

With this procedure we were able to see notable differences. In the first execution with patients containing basic values and insulin (3 patients), we could appreciate improvements:

For instance, with our patient tagged as Patient 8, we got the following results:

- Model with no training:

```
Zone A: 880 (76.78883071553228%)
Zone B: 246 (21.465968586387437%)
Zone C: 7 (0.6108202443280977%)
Zone D: 13 (1.1343804537521813%)
Zone E: 0 (0.0%)
```

RMSE: 32.800755

- Model pre-trained with other patients:

Zone A: 925 (80.7155322862129%)
 Zone B: 198 (17.277486910994764%)
 Zone C: 0 (0.0%)
 Zone D: 23 (2.006980802792321%)
 Zone E: 0 (0.0%)

RMSE: 23.619013

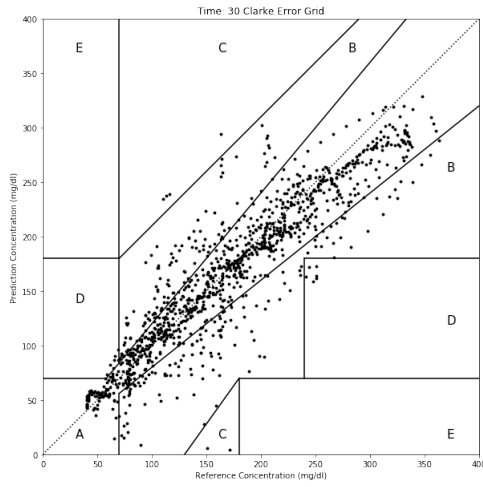


Figure 17: New FF model with glucose, basic vales and insulin

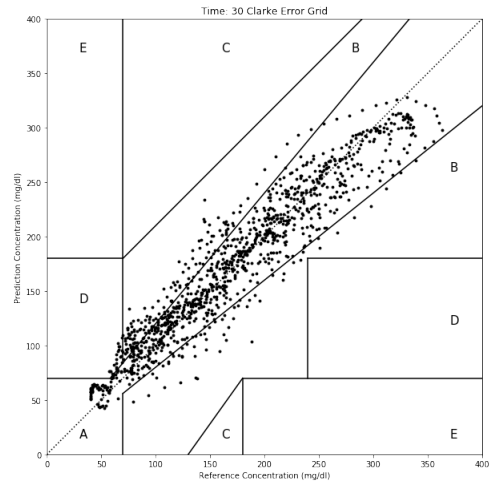


Figure 18: Pre-trained FF model with glucose, basic vales and insulin

As seen in both figures, the points in the figure 17 (no pre-trained model) are more scattered than in the figure 18 with the pre-trained model. We can appreciate that the pre-trained model tends to group the points. Overall, pre-trained models with different patients with insulin and basic values obtain more accurate results.

2. Model with basic values

Then, we proved with the same dynamic explained above to train and prepare a model with patients with only basic values (glucose, steps, heart rate and calories). This model was trained with 10 patients, so the expectations were higher.

The model had a notable improvement; when comparing all the Clarke

images generated in the notebook for 10 patients, data had visible changes and the goal to group the points in the zone A was almost accomplished.

Let's see an example with the patient tagged as Patient 5:

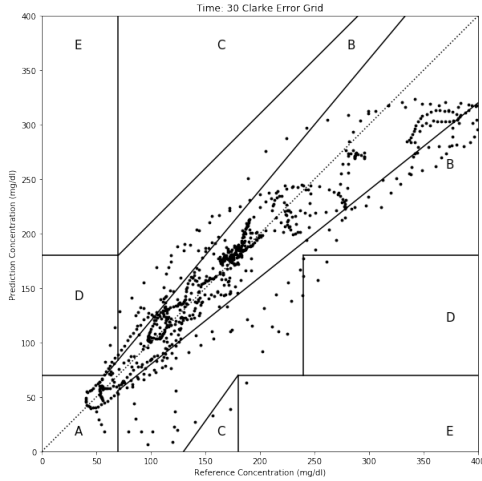


Figure 19: New model of glucose and basic values with FF

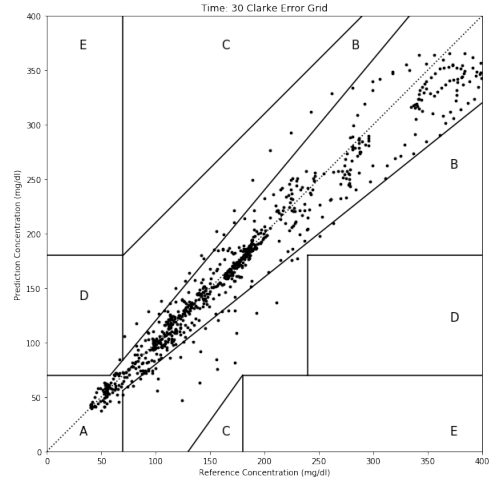


Figure 20: Pre-trained model of glucose and basic values with FF

In Figure 19 we can see that, with a new model, there are many scattered points, but with the pre-trained model (figure 20) the distribution of the points is really remarkable. The RMSE reduces its value from 36.3 to 20.8. The conclusion with this model is that the insulin value is not as relevant as the number of patients and data to train. Furthermore, these evidences confirm that it is possible to have good results with models trained with different patients when personal models are not available.

3. Model with glucose and insulin

Afterwards, we created a new model for the Ohio's Hospital patients in which, after pre-processing their data, we only kept the glucose and insulin values. The data-sets from these patients contain much more data than the Spanish patients. After training the model with all the patients, and comparing the efficiency of the model with the method

described before, we noticed that, with this data-set of patients, there is no improvements compared with the others. Also, we could appreciate that only a few have personal behaviours, so the results with a general model are almost the same. Furthermore, as in the others models, we figured out that the full-trained model has the tendency to increase the points that fall in zone A (Figure 21 and 22).

Example with our patient tagged as Patient 6 from Ohio's Hospital:

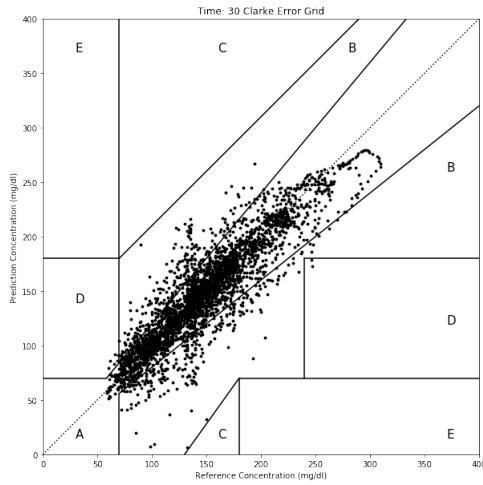


Figure 21: New model of glucose and insulin with FF

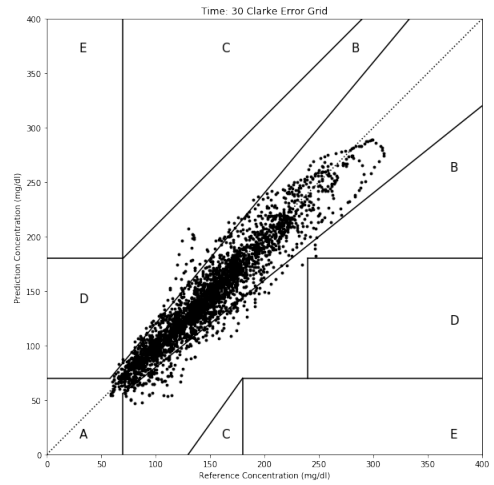


Figure 22: Pre-trained model of glucose and insulin with FF

RMSE new model = 19.19

RMSE pre-trained model = 14.13

4. Model with glucose

Finally, we created a general model that can be used for any patient, because it just requires glucose values. This model was trained with all the patients data-sets we had access to, except 3 patients that we reserved to test the model. Instead of comparing the results with a new model and the trained model for every patient, we are going to test the behaviour of the model with three patients that were not used to train the model. This means that we were predicting the glucose

values over the entire data-set of a patient instead of predicting only a test part of the patient (30% of the entire data-set). Hence, there are more values to predict and the probability of getting better results is lower. Nevertheless, the results with this three patients were robust and accurate. For example, the results for our patient tagged as Patient 14 are the following (figure 23):

Zone A: 3390 (88.88306240167803%)
 Zone B: 392 (10.27792343995805%)
 Zone C: 0 (0.0%)
 Zone D: 32 (0.8390141583639223%)
 Zone E: 0 (0.0%)

RMSE: 17.14621

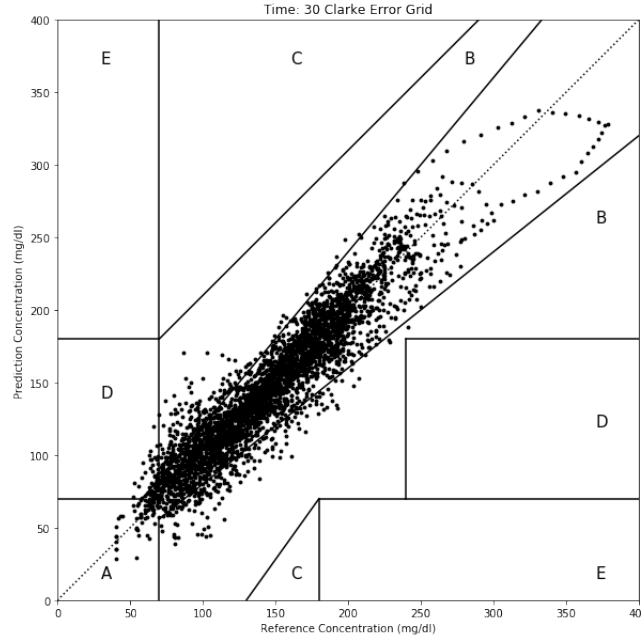


Figure 23: Patient 14 results with the pre-trained model with glucose values with FF

Overall, after testing all the different configurations, we gave more importance to the number of patients we trained rather than the number of parameters of the data-set. Based on the model with glucose, we can conclude

that the glucose is the 'mirror' of the other parameters, and the variations of each parameter is reflected in the glucose levels. Thus, if patients have a normal behaviour in a regular day, they would obtain consistent predictions.

After testing and explaining in detail all the different models with a 30 minutes forecasting, we are going to show the results of the glucose prediction for 60, 90 and 120 minutes. In this case, we are presenting only the glucose model, which returns the best results. We are using our patient tagged as Patient 14 to compare results:

- Results for the glucose prediction of 60 minutes (Figure 24).

Zone A: 2713 (71.24474789915966%)
Zone B: 976 (25.630252100840334%)
Zone C: 0 (0.0%)
Zone D: 119 (3.125%)
Zone E: 0 (0.0%)

RMSE: 29.09

- Results for the glucose prediction of 90 minutes (Figure 25).

Zone A: 2356 (61.96738558653341%)
Zone B: 1272 (33.45607574960547%)
Zone C: 0 (0.0%)
Zone D: 173 (4.550236717517096%)
Zone E: 1 (0.026301946344029457%)

RMSE: 36.83

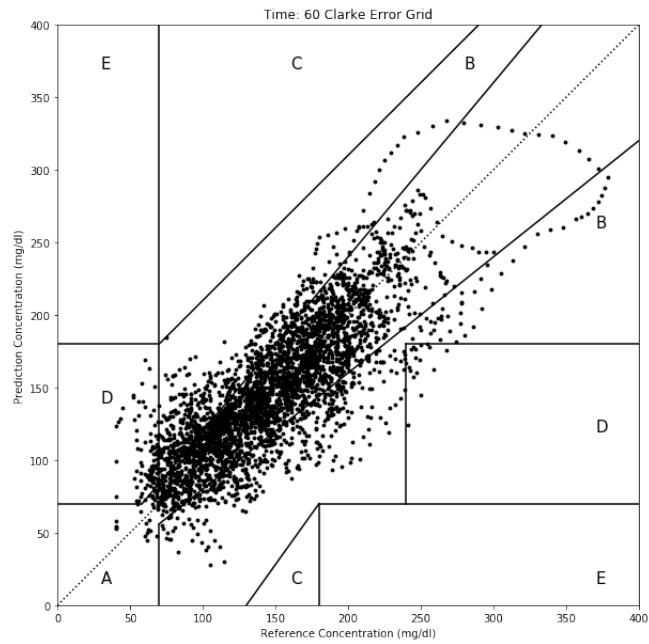


Figure 24: 60 minutes prediction with a pre-trained model of glucose values with FF

- Results for the glucose prediction of 120 minutes (Figure 26).

Zone A: 1424 (37.71186440677966%)
 Zone B: 1835 (48.59639830508475%)
 Zone C: 161 (4.263771186440677%)
 Zone D: 337 (8.92478813559322%)
 Zone E: 19 (0.5031779661016949%)

RMSE: 42.63

As seen in the graphics, the critical zones where the values of glucose fall tend to return worse results the longer the time is.

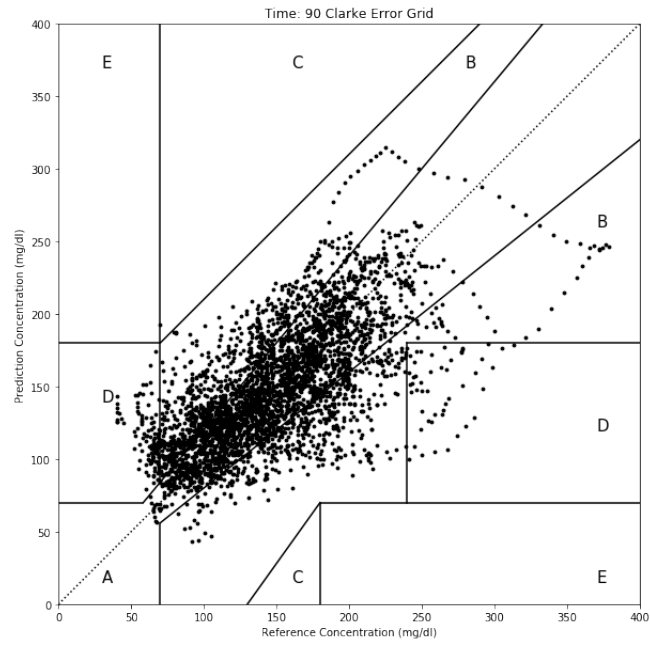


Figure 25: 90 minutes prediction with a pre-trained model of glucose values with FF

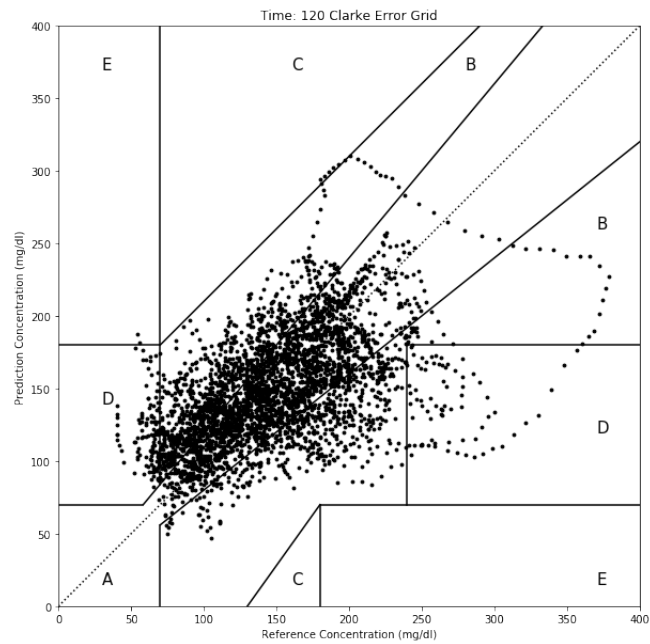


Figure 26: 120 minutes prediction with a pre-trained model of glucose values with FF

5.2.4. Train with LSTM

To test the efficiency of the model, we are going to use the same dynamic as in the previous section (section 5.2.2), and we are going to compare the results of this model with the previous configuration of the model.

The file in which all these results and studies are stored is 'Model with LSTM.pynb'.

1. Model with insulin and basic values

After training the new model with the three available patients with insulin and basic values, the first appreciable aspect is the time of training . Each training lasts for around 15 seconds compared to the 3 seconds of the Feed Forward model.

Lets compare the RMSE between the Feed Forward model and LSTM model:

- Patient tagged as Patient 7: FF: 31.037289 LSTM: 30.271496
- Patient tagged as Patient 8: FF: 24.921347 LSTM: 22.1213
- Patient tagged as Patient 9: FF: 23.814802 LSTM: 21.854898

As seen in the comparison, the LSTM model gives more accurate predictions. Furthermore, the tendency of the FF model to rise the points is omitted in this type of models. The predicted values seem to be corrected during the training without a deterministic rule, working on each of the risk zones.

Figure 27 shows the results with the LSTM model, and Figure 28 the results with the FF model.

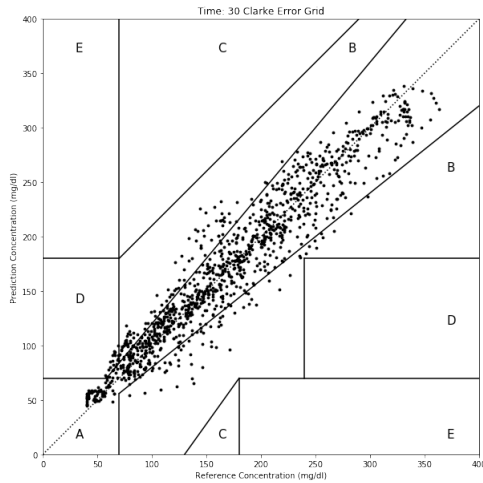


Figure 27: Pre-trained model of glucose, basic values and insulin with LSTM

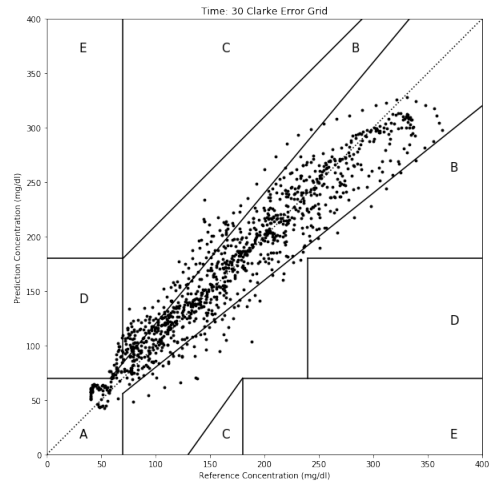


Figure 28: Pre-trained model of glucose, basic values and insulin with FF

2. Model with basic values

Now let's examine the performance of a LSTM model with more data to train. In this case, 10 patients with only the basic values (glucose, heart rate, steps and calories) were used. The results of the the RMSE model compared to those of the Feed Forward model are really similar. However, the Clarke images generated are in general more precise with the LSTM model.

As seen in the previous model with insulin, the behaviour in grouping the points with this model is less radical. It seems that the way predictions are improved is more personalized on the type of error, not having the tendency to up all points. Let's see an example with patient tagged as Patient 5 (new model results are presented in Figure 29 and pre-trained model results in Figure 30):

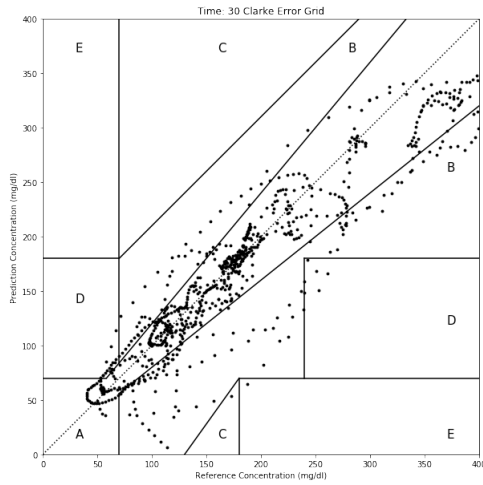


Figure 29: New model of glucose and basic values with LSTM

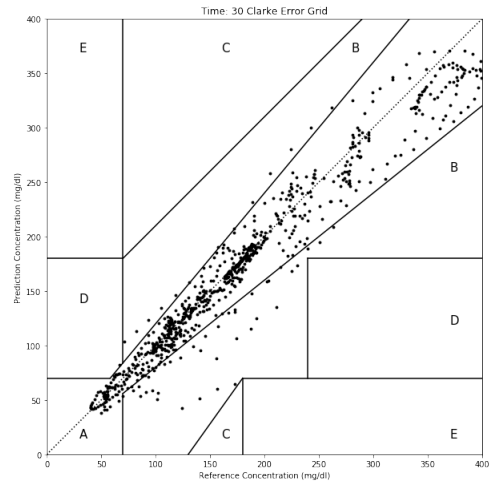


Figure 30: Pre-trained model of glucose and basic values with LSTM

3. Model with glucose and insulin

Overall, the results with this model are better than with the Feed Forward one, both the RMSE value and in the Clarke image. However, it's important to mention that a patient (patient tagged as Patient 5) of this set had a peculiar conduct of his glucose level that stands out over the others; that's the reason why using a new model for this patient returns better results than using a pre-trained model. Nonetheless, the results of both models are really similar.

Results of patient tagged as Patient 6:

With new model:

Zone A: 3665 (89.9607265586647%)
 Zone B: 396 (9.72017673048601%)
 Zone C: 0 (0.0%)
 Zone D: 13 (0.31909671084928815%)
 Zone E: 0 (0.0%)

RMSE: 16.55

With pre-trained model:

Zone A: 3849 (94.47717231222386%)
 Zone B: 217 (5.326460481099656%)
 Zone C: 0 (0.0%)
 Zone D: 8 (0.19636720667648502%)
 Zone E: 0 (0.0%)

RMSE: 13.29

4. Model with glucose

Finally, after training all the patients only with the glucose parameter, the results are a bit better than with the Feed Forward model, but nothing remarkable.

Results for patient tagged as Patient 14 (Figure 31):

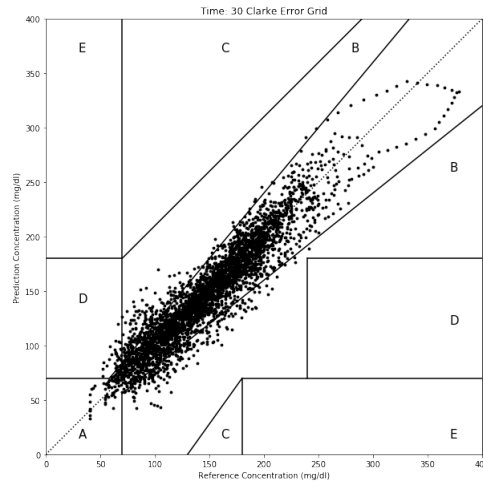


Figure 31: Results of testing Patient 14 with our 30 min LSTM model that works only with glucose values

Zone A: 3394 (88.98793917147351%)
 Zone B: 388 (10.17304667016256%)
 Zone C: 0 (0.0%)
 Zone D: 32 (0.8390141583639223%)
 Zone E: 0 (0.0%)

RMSE: 17.001036

The conclusion after comparing these two models is that the Feed Forward model has a particular behaviour and tendency so, at the time of training a particular patient that has a data-set that stands out over the others, the predictions are worse. On the other hand, the LSTM adapts to the type of patient, making it more reliable and precise.

After testing and explaining in detail all the different models with a 30 minutes forecasting, we are going to show the results of the glucose prediction for 60, 90 and 120 minutes. In this case, we are presenting only the glucose model, which returns the best results. We are using our patient tagged as Patient 14 to compare results:

- Results for the glucose prediction of 60 minutes (Figure 32):

Zone A: 2760 (72.47899159663865%)
Zone B: 922 (24.212184873949578%)
Zone C: 0 (0.0%)
Zone D: 126 (3.308823529411765%)
Zone E: 0 (0.0%)

RMSE: 28.71

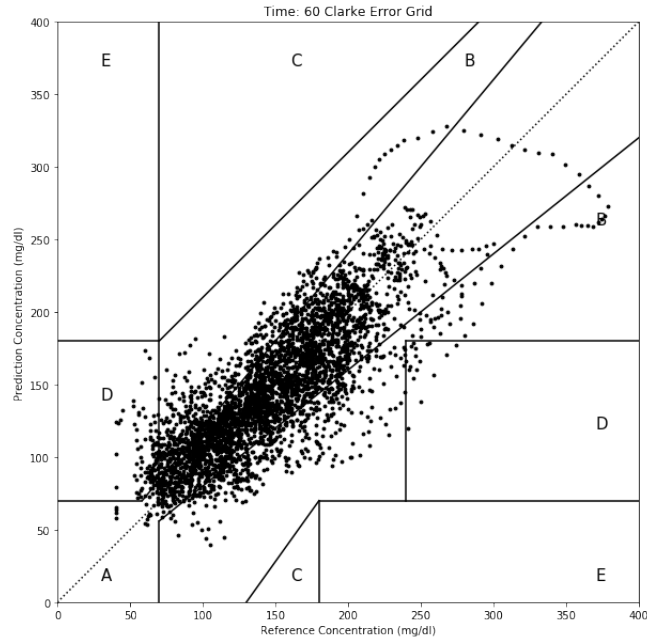


Figure 32: Results of testing patient 14 with our 60 min LSTM model that works only with glucose values

- Results for the glucose prediction of 90 minutes (Figure 33):

Zone A: 2245 (59.047869542346135%)
 Zone B: 1387 (36.48079957916886%)
 Zone C: 8 (0.21041557075223566%)
 Zone D: 160 (4.208311415044713%)
 Zone E: 2 (0.052603892688058915%)

RMSE: 38.74

- Results for the glucose prediction of 120 minutes (Figure 34):

Zone A: 2081 (54.82086406743941%)
 Zone B: 1496 (39.409905163329825%)

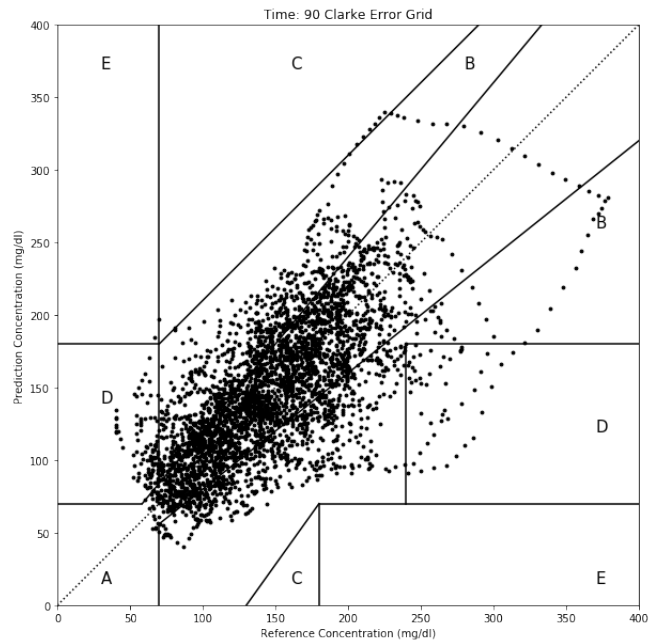


Figure 33: Results of testing patient 14 with our 90 min LSTM model that works only with glucose values

Zone C: 6 (0.15806111696522657%)
 Zone D: 211 (5.558482613277134%)
 Zone E: 2 (0.052687038988408846%)

RMSE: 42.56

Overall, the results compared to the Feed Forward are really similar, but the training time of all this models is much longer.

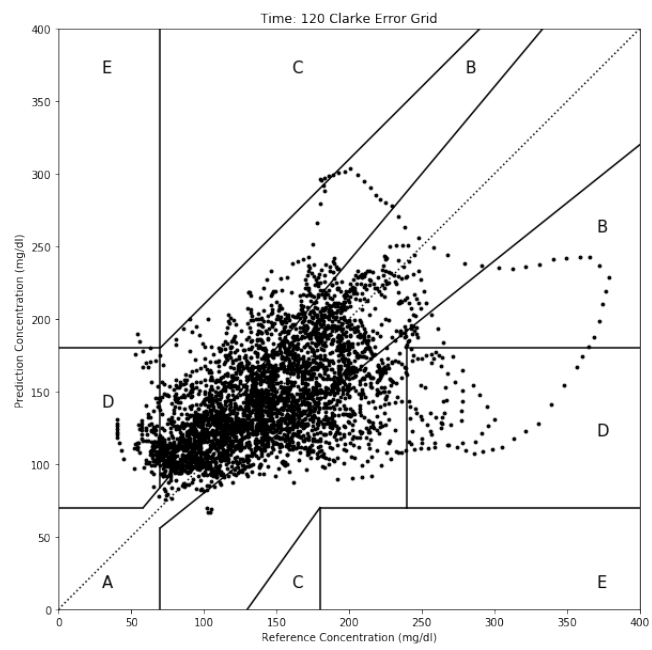


Figure 34: Results of testing patient 14 with our 120 min LSTM model that works only with glucose values

5.2.5. Train with NVIDIA Digits

Given the fact that all the different configurations and NN returned similar results, we are going to comment one of them without specific details of the configuration. The NN used for this report is AlexNet network with Adam (Adaptive Moment Estimation) solver / RMSprop, training during 50 epochs with a base learning rate of 0.5.

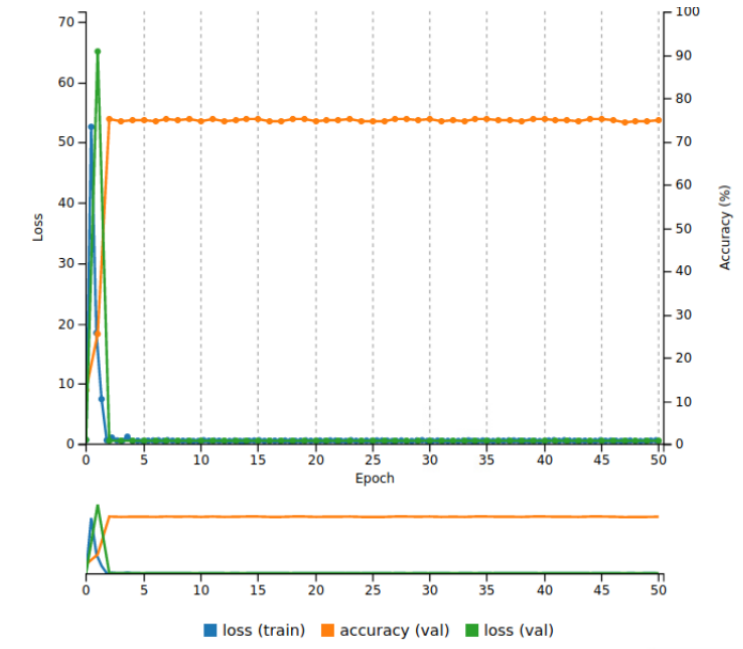


Figure 35: Process of training with all the images on Nvidia Digits

In first instance, we saw very promising results with around 78% of accuracy. However, when we went into further detail, we noticed that the accuracy remained always around the same value during the whole training. To prove the accuracy of the neural net, we tried 2 different predictions from an Hypo class image and a Normal class image. For both images the result was the same, 78% of Normal in prediction. From this predictions we could observe that the neural net had a statistic influence, 78 % of the images are Normal. To prove that the neural net only used statistical criteria for prediction, we decided to train another neural network with more balanced data input, 64% of Normal images and 36% of Hypo images.

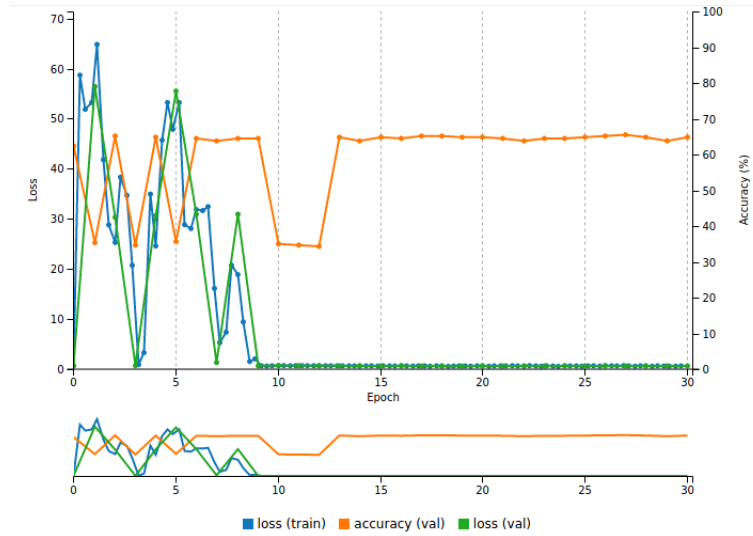


Figure 36: Process of training with image-data balanced on Nvidia Digits

This last result, shown in Figure 36 could confirm our first hypothesis: the NN is not able to find any characteristic to differentiate between the two classes.

There is still a lot of work to do until we reach good results. After our work, we can see two clear ways to find a functional model.

- Explore other ways to represent the data with images.
- Try out different NN, pre-trained models focused on medical data analysis and apply other techniques such as transferred learning.

6. Conclusions and Future work

6.1. General conclusions

After having carried out the creation of a NN model to predict glucose levels and a website service where we can share our algorithm with patients of diabetes of type 1, we feel satisfied to have developed a first version of a final service that encompasses both a research work of glucose prediction (covering different scopes in this sector) and an intuitive web interface for the user.

In the stage of pre-processing, we learned the different behaviours of diabetes patients and we familiarized with their levels of glucose, discovering which parameters were the best for an accurate prediction.

We also realized that much of the patients parameters given were not really necessary to obtain good predictions of glucose, so basic values as the glucose, steps and few more parameters easy to measure are enough to get a precise result. Therefore, if the patient has an appropriate system to measure the levels of glucose, he will be able to obtain good predictions.

Also, as seen on the results report, one concept we learned with this study is the following quote: *'the glucose is the 'mirror' of other diabetes parameters, and the variations of each parameter are reflected in the levels of glucose'*. Hence, if the patient has a regular conduct related to physical exercise, food..., the patient should obtain an accurate prediction based only on the evolution of its levels of glucose.

After testing all the models implemented with Keras, the idea of predicting glucose behaviour with images came up (J. Ignacio Hidalgo's idea). We were inexperienced in this type of training using the NVIDIA Digits and, although we didn't obtain good results because it's a new field that needs further research, we acquired some knowledge about image training and types of representations for the diabetes parameters. Also, we discovered that it is a really good system to return prediction results, because this service has a

really intuitive and visual interface. However, as we were more familiar with the Python scripts and we were incapable of obtaining safe and reliable results with NVIDIA Digits, we decided to use the developed models in Python.

Another aspect to highlight is that this full project was realized at the distance, because both of us had an Erasmus studentship, and the hole project was developed through video calls from Leiden, Netherlands (Alvaro's destination) and Milán, Italy (Alejandro's destination). Also we had weekly calls with our tutor J. Ignacio Hidalgo, and we didn't experience problems and realized that it is possible to work far from each other.

Overall, we acquired valuable knowledge related to diverse fields of computer science such as: data from diabetes patients, correlation of data with glucose, types of predictions for glucose forecasting, how to create a website with Django and incorporate the implemented scripts, how to connect a SQL database to save the information, etc. In conclusion, we consider that this final degree project is the result and representation of four years of learning.

6.2. Future work

Since the approach of the project is very general and the main objective was to create a fully working online application that provides useful information to patients, we could not get deep on each one of the task involving GlucNet.

Obviously, there is a lot of investigation remaining on the structure of the neural nets. With the existing technologies, different libraries and types of NN it is not possible to try out every options. Anyways, this is the work that needs to be done now on, with the support of the current website. Also, other techniques in the glucose prediction field can be studied in order to give them support on GlucNet.

In the future, the idea is to apply different forecasting and ensembles techniques such as KNN, random forest, gradient boosting... and compute more metrics to have a better evaluation of the results.

Furthermore, the training and classification of wavelets images is a field which needs time to be studied to accomplish reasonable results. Also, in the future, the idea is to add the results of NVIDIA Digits to the website

service, and display the results with the current ones.

Relating to the website, a lot of improvements can still be done. Our main goal for the future will be to improve the user experience, giving the user more features to work with such as better results display, more control on personal models, easier way to handle the data... in a simple environment that hides the difficulty of working with Neural Networks.

7. Alejandro Varela work

In this section I'm going to relate my work during this final degree project. First of all, I want to emphasise that the full project was developed by both of us cooperating with our professor J. Ignacio Hidalgo. Even though we divided certain parts, at all times we were conscientious of the work of each, and mostly this project was done working side by side through video-calls using different apps, like Discord, Skype, Meets, etc. I want to highlight that this is not my first time working with Alvaro; he has been my degree partner these four years, and we have lot of experience working together.

To start with this degree project, we had several video-calls with our professor to get familiar with the diabetes pathology and current forecasting techniques that are used or have been used. Once we acquired this knowledge about glucose predictions, metrics, algorithms and the pathology itself, I felt really motivated because, after discovering the machine learning field in second grade, it was the first time I could apply my knowledge in a real problem.

After having clear the goals of the project, the first step was to study our first data-set obtained from a Spanish patient. We worked together in this part because it was relevant to understand the parameters. Due to we took the 'Machine learning' subject in third grade, and we also did the project of the subject together, we applied some pre-processing techniques learned to investigate the correlation between parameters and select the best data configuration. Then we started to pre-process the data filling missing values, transforming the data-set to supervised as explained in section 4.3.4. This was our first time working with time series and we learned a lot about this.

Afterwards, we set up a first simple model and obtained our first results. Then we applied to the results the different metrics discussed in the section 5.1 and realized that the first execution was really hopeful. At that point, we were very motivated to continue testing different techniques.

Then we had a meeting with our professor where he could see that the results were promising, so he saw convenient to distribute us more data-set of different patients from Spain and Ohio. Taking into account that the Ohio patients contained different parameters than the Spanish ones, we started to divide our work to optimize the realization of the tasks. Alvaro took care of

preparing the Ohio patients, and I worked on automating the pre-processing of the data. I created a script with functions that cut the data-set in the parts which contain data, interpolate missing values, and adapt the functions implemented in the first execution for all the possible patients. Also, I studied the rest of the Spanish patients that J. Ignacio Hidalgo gave us, and get some conclusions together. Again, all this work was done in a video-call with Alvaro and every time we were cooperating.

Soon afterward, the professor came up with the idea of making glucose forecasting with image classification. Thus, the course of the project changed, and we started to learn image classification, conversion of numerical data to images, and the prediction of glucose would change from 30,60, 90 and 120 minutes glucose forecasting to identify if the patient could have hypoglycemia the next day. We collaborated with Jose Manuel Velasco, who taught us how to generate a wavelet image from glucose values, and the features of that type of images. First we trained and obtained results with images that Alvaro and I created with the numerical data just by plotting the values in a graphic, and we got in touch with NVIDIA Digits.

As a result of the image training, we realized that this field of glucose training will have hopeful results, but needs much research and time. So we resume our previous work in Python and numerical data. Due to we only trained and tested one patient, we created two scripts: 'Model dense layer (original FF)' and 'Model with LSTM', where all the patients were trained and tested with both types of models.

Once we created all the possible models depending on the type of data, we started working together on the web. We created our first screen and got familiar with the Django framework. Then I got more focused on adapting the Python scripts done in the last months to couple them in the back-end of the web. I also prepared the front-end part and tried my best for having a simple and aesthetic design. I also putted in work the database that we previously designed and incorporated to the website.

Lastly, we started to write this memory for reporting nine months of work where I have learned a lot about this type of predictions. This project has helped me to guide myself in my professional career, and to top off this four years of constant learning.

8. Alvaro Delgado work

First of all, I want to note that almost all the work for this project have been done by both of us in parallel, even when we divided the work we kept working online through video-calls, having the chance to report each other the ideas and doubts that came out to our minds. While working, each of us from our respective Erasmus destinations, we made use of technologies such as Skype, Google Meets, Discord and Github to have full knowledge of the work we did independently.

From the start of the Final Degree Project, a work of investigation have been done in order to get knowledge about the new technologies I did not know. This covers learning how to properly use the tools, mainly from official sources such as NVIDIA Digits user manual, Django user manual, several reports about graphic NN and glucose forecasting and database implementation. Since our FDP tutor, J. Ignacio Hidalgo, is a very experimented investigator in the field of Glucose forecasting for diabetes patients, talking with him about the different approaches of the project have been one of my main sources of learning all along the project.

When we talk about actual implementation work, it started with studying the data-set, trying different configurations of it along with adapting it to supervised learning. After we prepared the data-set to be supervised, the first neural network was created to get our first results. After this first execution, we noticed that our data was prepared enough and we started to get deeper into the Neural Networks.

Once the first model was implemented and tested, our professor distributed us more patients for having more data to train and obtain better results. He provide us new type of patients from the Hospital of Ohio with different parameters. I created a script for pre-processing this type of data, and select the most optimal configuration for this patients.

At this point, we worked several hours testing different NN and representations of the results, getting clearer of how we were going to implement the final solution.

When we got our Feed Forward model that gave us the first valuable

results, we investigated on the use of other NN. Then we decided to try out another NN with the LSTM type. In order to prepare the code for the website, we started creating the first scripts that automatize the process of training, those are 'Model dense layer (original FF)' and 'Model with LSTM'.

After this, we decided to start splitting the tasks in order to move faster. Thus, I started to investigate more about Graphical NN and how to represent the data to feed these NN. An special mention is required since this innovative approach was an idea given by our tutor.

A long work of trying different image representations was done, going from simple linear representations to continuous wavelet transformations. This work was done in R. I used all these representations to feed the NN implemented in NVIDIA Digits, looking for the best way.

Once we decided that the graphic NN were not strong enough to create an application with an acceptable fidelity, we chose the numeric NN for the final solution.

At this last part of the project, we divided the work again for the website implementation. My job was to create the back-end of the website, including the implementation of the scripts that make the training and prediction in the website, different button functionalities and other aspects of the website engine.

Finally, when we got the project finished, we started writing this final report together. This was a constant work of writing a section independently and then, switching the parts to have a second review.

9. Acknowledgment

Finally, it only remains to mention that we are very grateful to have the opportunity that the professor J. Ignacio Hidalgo has given us. Thanks to his constant dedication for this project, we acquire much experience in this sector of Computer Science, and we have always been in safe hands of professional and experienced persons as our professor, Jose Manuel Velasco and the rest of AbsysGroup.

Also we have to mention and thank the Universidad Complutense de Madrid for this four years of learning. After having studied both abroad, we agree that we are very fortunate for being part of the UCM family.

Thanks to our families that give us the opportunity and the support to study a degree, something that not all the people can afford.

Thanks to this FDP, now we have clearer ideas about the field we are interested in.

Thanks to all the people that have been in our path to here, you all had part in this project.

Only when you have walked the road, you know how good it was the way.

10. Bibliography

- Tensor-flow user guide
- Keras user guide (<https://keras.io/guides/>)
- Nvidia Digits user guide
- Django user guide
- <https://runder.io/optimizing-gradient-descent/index.html>
- <https://diatribe.org/understanding-average-glucose-standard-deviation-cv-and-blood-sugar-variability>
- <https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464>
- Stackoverflow
- Multiple other sources

References

- [1] J. I. Hidalgo, J. M. Colmenar, G. Kronberger, S. M. Winkler, O. Garnica, J. Lanchares, Data based prediction of blood glucose concentrations using evolutionary methods, *J. Medical Systems* 41 (2017) 142.
- [2] C. Pérez-Gandía, A. Facchinetti, G. Sparacino, C. Cobelli, E. Gómez, M. Rigla, A. de Leiva, M. Hernando, Artificial neural network algorithm for online glucose prediction from continuous glucose monitoring, *Diabetes technology & therapeutics* 12 (2010) 81–88.
- [3] V. Tresp, T. Briegel, J. Moody, Neural-network models for the blood glucose metabolism of a diabetic, *IEEE Transactions on Neural networks* 10 (1999) 1204–1213.
- [4] F. Allam, Z. Nossai, H. Gomma, I. Ibrahim, M. Abdelsalam, A recurrent neural network approach for predicting glucose concentration in type-1 diabetic patients, in: *Engineering Applications of Neural Networks*, Springer, 2011, pp. 254–259.

- [5] Q. Sun, M. V. Jankovic, L. Bally, S. G. Mougiakakou, Predicting blood glucose with an lstm and bi-lstm based deep neural network, in: 2018 14th Symposium on Neural Networks and Applications (NEUREL), IEEE, 2018, pp. 1–5.
- [6] E. M. Aiello, G. Lisanti, L. Magni, M. Musci, C. Toffanin, Therapy-driven deep glucose forecasting, *Engineering Applications of Artificial Intelligence* 87 (2020) 103255.
- [7] T. El Idriss, A. Idri, I. Abnane, Z. Bakkoury, Predicting blood glucose using an lstm neural network, in: 2019 Federated Conference on Computer Science and Information Systems (FedCSIS), IEEE, 2019, pp. 35–41.
- [8] C. Meijner, S. Persson, Blood Glucose Prediction for Type 1 Diabetes using Machine Learning Long Short-term Memory based models for blood glucose prediction, Master’s thesis, 2017.
- [9] J. Martinsson, A. Schliep, B. Eliasson, C. Meijner, S. Persson, O. Morgren, Automatic blood glucose prediction with confidence using recurrent neural networks, in: 3rd International Workshop on Knowledge Discovery in Healthcare Data, KDH@ IJCAI-ECAI 2018, 13 July 2018, 2018, pp. 64–68.
- [10] S. Mirshekarian, R. Bunescu, C. Marling, F. Schwartz, Using lstms to learn physiological models of blood glucose behavior, in: 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), IEEE, 2017, pp. 2887–2891.
- [11] S. Mirshekarian, H. Shen, R. Bunescu, C. Marling, Lstms and neural attention models for blood glucose prediction: Comparative experiments on real and synthetic data, in: 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), IEEE, 2019, pp. 706–712.
- [12] M. Mayo, L. Chepulis, R. G. Paul, Glycemic-aware metrics and over-sampling techniques for predicting blood glucose levels using machine learning, *PloS one* 14 (2019).

- [13] K. Li, J. Daniels, C. Liu, P. Herrero, P. Georgiou, Convolutional recurrent neural networks for glucose prediction, *IEEE journal of biomedical and health informatics* 24 (2019) 603–613.
- [14] A. Bertachi, L. Biagi, I. Contreras, N. Luo, J. Vehí, Prediction of blood glucose levels and nocturnal hypoglycemia using physiological models and artificial neural networks., in: *KHD@ IJCAI*, 2018, pp. 85–90.
- [15] K. S. Eljil, G. Qadah, M. Pasquier, Predicting hypoglycemia in diabetic patients using time-sensitive artificial neural networks, *International Journal of Healthcare Information Systems and Informatics (IJHISI)* 11 (2016) 70–88.
- [16] X. Mo, Y. Wang, X. Wu, Hypoglycemia prediction using extreme learning machine (elm) and regularized elm, in: *2013 25th Chinese Control and Decision Conference (CCDC)*, IEEE, 2013, pp. 4405–4409.
- [17] P. P. San, S. H. Ling, H. Nguyen, et al., Evolvable rough-block-based neural network and its biomedical application to hypoglycemia detection system, *IEEE transactions on cybernetics* 44 (2013) 1338–1349.