MDPI

Article

Prediction of Glucose Concentration in Children with Type 1 Diabetes Using Neural Networks: An Edge Computing Application

Federico D'Antoni [1,‡,*], Lorenzo Petrosino [1,‡], Fabiola Sgarro [1], Antonio Pagano [1], Luca Vollero [1], Vincenzo Piemonte [2] and Mario Merone [1,*]

[1] Unit of Computer Systems and Bioinformatics, Department of Engineering, Università Campus Bio-Medico di Roma, Rome, Italy.
[2] Unit of Chemical Engineering, Department of Engineering, Università Campus Bio-Medico di Roma, Rome, Italy.
* Corresponding author: f.dantoni@unicampus.it (F.D.);m.merone@unicampus.it (M.M.)
‡ These authors contributed equally to this work.

Abstract: Background: Type 1 Diabetes Mellitus (T1D) is an autoimmune disease which can cause serious complications, that can be avoided by preventing the glycemic levels from exceeding the physiological range. Straightforwardly, many data-driven models were developed to forecast future glycemic levels, and to allow patients avoiding adverse events. Most models are tuned on data of adult patients, whereas the prediction of glycemic levels of pediatric patients has been rarely investigated, as they represent the most challenging T1D population. Methods: A Convolutional Neural Network (CNN) and a Long Short-Term Memory (LSTM) Recurrent Neural Network were optimized on glucose, insulin, and meal data of 10 virtual pediatric patients. The trained models were then implemented on two edge-computing boards to evaluate the feasibility of an edge system for glucose forecasting, in terms of prediction accuracy and inference time. Results: The LSTM model achieved the best numeric and clinical accuracy when tested in the *.tflite* format, whereas the CNN achieved the best clinical accuracy in *uint8*. The inference time for each prediction was far under the limit represented by the sampling period. Conclusion: Both models are effective in predicting glucose in pediatric patients in terms of numerical and clinical accuracy. The edge implementation did not show significant performance decrease, and the inference time was largely adequate for a real-time application.

Keywords: Diabetes; Time-Series forecasting; Glucose Prediction; Pediatrics; Edge Computing; Neural Network; Decision Support System; Precision Medicine; Artificial Intelligence

Citation: Lastname, F.; Lastname, F.; Lastname, F. Title. *Bioengineering* **2022**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Submitted to *Bioengineering* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Type 1 Diabetes Mellitus (T1D) is a chronic disease in which the pancreas produces little or no insulin. If not treated properly, it can lead to both short- and long-term complications, including micro- and macro-vascular diseases that can damage kidneys, eyes, liver, and the circulatory system [1]. Although T1D has no cure, it can be managed through daily insulin administrations, with the aim of keeping the glycemic level in the euglycemic range, i.e. between 70 and 180 mg/dl. In recent years, the utilization of Continuous Glucose Monitoring (CGM) devices increased consistently, because these devices allow patients to keep track of their glycemic trend 24 hours a day.

The quality of life of people suffering from T1D improves considerably by preventing the blood glucose levels from exceeding the euglycemic range [2]. Albeit CGM devices have greatly enhanced the management of the disease [3], frequent hyperglycemic ($CGM > 180\,mg/dl$) and hypoglycemic events ($CGM < 70\,mg/dl$) are reported in clinical data. For this reason, in the last decade many mathematical models have been developed with the aim of predicting future glucose levels [4]. Indeed, an accurate forecast of the future

glycemic level allows the patient to adjust their therapy in order to prevent undesirable events. In particular, after having fixed a prediction horizon (PH), i.e. how forward in time the prediction is made, such models exploit the recent trends of CGM and other features such as the injected insulin in order to predict, through the medium of a regression task, what the glycemic level will be after PH minutes.

Although many physiological-based mathematical models exist for the prediction of future glycemic levels [5,6] the vast majority of recent research moved towards the implementation of data-driven models. In the latter case, whether machine learning and neural network/deep learning models have been implemented, and the networks generally achieve better results [7]. Also, some models capable of updating their training in order to catch more recent variations of the glycemic trend have been proposed [8]. The most widely used performance evaluation metric for blood glucose levels forecasting is the Root Mean Square Error (RMSE), that will be defined formally in section 2.2. Briefly, the smaller the value, the better the performance.

In the frame of machine learning techniques, Bunescu et al. [9] use a three-compartmental physiological model of blood glucose dynamics to generate features for a Support Vector Regressor (SVR) that is trained on patient specific data. The model is validated on data of 5 T1D patients from a private dataset. The blood glucose levels forecasts with a 30- and 60-minute PH attain RMSE values equal to 22.6 mg/dl and 35.8 mg/dl, respectively. Georga et al. [10] present a Random Forest regression technique for the personalized prediction of the glucose concentration in T1D patients. This multivariate model takes as input CGM data, physiological features and lifestyle information. High-accuracy forecasts are derived for a 15-minute PH if all the available features are used (RMSE $= 6.6 \pm 1.3\, mg/dl$), whereas the performance considerably deteriorates when exploiting CGM data alone as input feature (RMSE $= 11.3 \pm 2.2\, mg/dl$). Sparacino et al. [11] propose a first order AR model with time-varying parameters, which are estimated at each time stamp using recursive least squares. They test several values of the forgetting factor with 30- and 45-minute prediction horizons. The model is tuned on CGM data of 28 T1D patients from a private dataset. Results are accurate enough to potentially avoid or mitigate critical adverse events (RMSE $= 18.3 \pm 11.8$ and $34.9 \pm 21.3\, mg/dl$).

In the frame of neural networks and deep learning techniques, several well-established models have been applied to the task of glycemic prediction, achieving the best performance in the literature, and some brand new models have been proposed from scratch for this specific task [12,13]. Mosquera-Lopez et al. [14] present a Long Short-Term Memory (LSTM) recurrent neural network with a correction module to predict glycemic levels with a PH of 30 minutes, tuning the model on data of more than 4000 patients from a private dataset and testing it on data of further 10 patients, achieving an average RMSE $7.6 \pm 2.2\, mg/dl$. Li et al. [15] propose a recurrent Convolutional Neural Network (CNN) to predict glycemic levels on simulated patients from the UVA/Padova simulator [16] and on 10 patients from private dataset with a PH of 30 and 60 minutes. They achieved better results for the simulated dataset (average RMSE$= 9.4 \pm 0.7\, mg/dl$ and $18.9 \pm 2.5\, mg/dl$) whereas performance degrades when testing on real data (RMSE$= 21.1 \pm 2.4\, mg/dl$ for 30 minutes, $33.3 \pm 4.8\, mg/dl$ for 60 minutes).

Despite the large amount of works presented for the forecasting of future glycemic levels and the noteworthy results they achieve, all the aforementioned papers focus on the prediction of glycemic levels of adult subjects. Indeed, there are few works in the literature that aim to predict blood glucose levels specifically in pediatric patients. Children represent the most challenging diabetic population, because pediatric patients go through a period of rapid growth, physiological and hormonal changes along with complex individualization and socialization processes. This often results in a significant decline in the quality of disease management, treatment adherence, and glycemic control [17,18]. Among the most remarkable studies, Mougiakakou et al. [19] test 2 different neural network models on real data of 4 T1D pediatric patients, after pre-processing features with a glucose-insulin metabolism model. They achieve the best results (average RMSE $= 22.2 \pm 13.4\, mg/dl$)

using a feedforward neural network. Dassau et al. [20] propose a hypoglycemia prediction algorithm that combines 5 different predictors to assess the risk of incoming hypoglycemia in the following 35 minutes on children with T1D, validating the system on 22 subjects. The decisions of the 5 models are combined through a majority vote, and the ensemble model identifies with sufficient advance 91% of the hypoglycemic events. Finally, De Bois et al. [21] test 6 different data-driven models on data of 10 virtual T1D children generated using the UVA/Padova simulator [16]. They generated for each patient 29 single days with a 3-meal daily scenarios, exploiting the simulator's built-in bolus calculator and treating each day as a standalone set of data. For a PH of 30 minutes, they achieve the best numerical performance using a Gaussian Process with dot-product kernel (average RMSE $= 5.2 \pm 2.0\, mg/dl$). Conversely, the LSTM model results the one with the greatest clinical accuracy, as 97.46% of its predictions fall into the zones A and B of the Clarke Error Grid [22]) corresponding to accurate predictions.

Normally, machine learning techniques are validated on laboratory setup, and, when they are applied in practice, they are performed directly on servers or centralized processing units. The task of future glycemic levels prediction makes no exception, as most systems that perform real-time prediction exchange data between an edge device, only used to gather information, and the cloud, where the actual glucose level forecasting is performed [23,24]. This is mainly due to the memory limits of edge-computing devices. Nonetheless, the drawback of such systems is that they constantly require an internet connection to work; this is not arguable with regards to medical devices, because an interruption in the signal may result in missing decision support to the user. However, the increasing development of new, more powerful and dedicated hardware, combined with the widespread use of IoT (Internet of Things) tools, is enabling the emergence of a branch of artificial intelligence known as inference at the edge [25,26]. This involves the machine learning models being run directly from a proximity device using data collected from associated sensors. Taking into account also the increasingly telemedicine-oriented approach [27,28], it becomes clear that the possibilities given by inference at the edge can be exploited to create predictive models that work in real-time with patient data to both improve the patients' life and increase the ability of the physicians to extract useful information from the sensor data. Compared to systems that run on the cloud, edge computing can provide more reliable real-time service, with low latency, and they are not limited by internet connectivity. For this reason, a recent study by Zhu et al. [29] proposed an Embedded Edge Evidential Neural Network to predict future glycemic levels of adult T1D patients in real time exploiting CGM sensor readings and an edge-computing device. Due to limitations in the computational capacity, they converted their TensorFlow model to C, and achieved an RMSE of 18.9 mg/dl with a PH of 30 minutes on both a public and a private dataset.

In the light of what is present in the literature, the contribution of this work is twofold. On the one hand, we implement two stat-of-the-art models for the prediction of glycemic levels, and apply them to the specific task of the prediction in pediatric patients; such models improve the performance of the models currently studied in this field. On the other hand, we implement these models on an edge computing system, thus laying the foundations for the future creation of embedded devices capable of forecasting blood glucose levels in order to improve patients' quality of life and aid medical diagnosis; we evaluate the feasibility of such prediction-at-the-edge system on two different boards in terms of prediction accuracy and execution time. To the best of our knowledge, this is the first attempt to implement a pediatric-specific glucose prediction model on an edge-computing system.

2. Materials and Methods

In this section, we present the generatd dataset utilized to tune the predictive models, the description of the hardware that we used as edge system for tests, and the experimental setup adopted with regards to the optimization of the neural network models as well as their implementation on the edge system.

2.1. Dataset

Data were produced for 10 pediatric patients by running several simulations in the UVA/Padova simulator [16]. Such a tool allows to generate different scenarios for *in silico* patients by only providing a meal schedule. The simulator is able to determine the optimal insulin boluses to be injected for each specific patient, and can thus provide the glycemic evolution for each subject for a pre-set number of days. However, the tool allows the user to modify the insulin bolus value, and to include a sensor error in the CGM readings. Data are generated with a 1-minute sampling.

Two different datasets were generated on a scenario consisting of 30 days of simulation with 5 meals per day. The first scenario has no errors in sensor reading and insulin administration, as automatically computed by the simulator, and thus corresponds to an ideal T1D management. Differently, we created the second scenario by including CGM sensor errors and by forcing the presence of hyperglycemic and hypoglycemic events. We were able to achieve such a goal by first allowing the UVA/Padova simulator to run a simulation with its own optimal bolus control; then, we extracted the vector of injected boluses and added random noise taken from a uniform distribution. In particular, each bolus consisting of I insulin units was modified according to the following:

$$\hat{I} = I + z \tag{1}$$

where z is a random value taken in the interval $[-3, 3]$. In practice, each bolus was increased or decreased by no more than 3 units of insulin from its optimal value. The modified bolus vector was given as effective bolus vector to the UVA/Padova to run the simulations for this scenario. This makes such a scenario more realistic, because in real life the increase or decrease in blood sugar levels occurs mainly due to an inaccurate estimate of the amount of carbohydrates ingested, or to deviations in correction dosing [30]: we added noise on insulin boluses to simulate human error.

The datasets consist of information on blood glucose levels and data on insulin (bolus, basal, and injection were added together and considered as a one) and finally carbohydrate intake. Specifically, the final datasets consider Insulin-On-Board (IOB) as insulin feature, which was manually generated by exploiting a mathematical model [31]. IOB is a quantity referred to the amount of rapid-acting insulin still active in the patient's body after a bolus injection, and thus provides deeper information on the recent history of insulin injections compared to the punctual insulin values themselves. The range of time for considering insulin still active is roughly between 2 and 8 hours [32]. IOB is estimated differently among the main insulin pump companies, but in all cases its calculation is based on insulin action plots which forecast the percentage of residual insulin as a function of time. For the Insulet pump, which is the one considered by the simulator, the active insulin time is equal to 3 hours and the shape of insulin action plot is linear [31]. Thus, the value of IOB for each timestamp t was computed as:

$$IOB(t) = \sum_{i=0}^{179} \alpha(i) u(t-i) \tag{2}$$

where $u(t-i)$ represents the insulin injection at timestamp $t-i$, and $\alpha(i) = 1 - i/180$ is the coefficient corresponding to the insulin decay curve. It is worth noting that only past insulin values (i.e., corresponding to timestamps $\leq t$) are used to compute the IOB. Specifically, 100% of the latest insulin injection value contributes to $IOB(t)$, whereas the contribution linearly decreases to 0 for older values in the previous 3 hours. Straightforwardly, the first 3 hours of data of each patient were not used to train the predictive models, as they were used to initialize the IOB values.

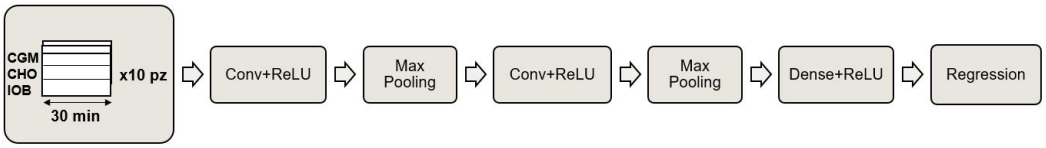

Figure 1. Schematic representation of the proposed Convolutional Neural Network.

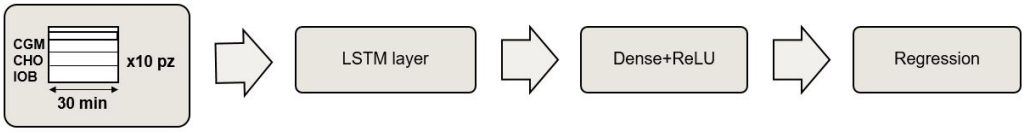

Figure 2. Schematic representation of the proposed LSTM Recurrent Neural Network.

2.2. Optimization of network models

A Precision Medicine approach was used to tune the predictive models, which involves choosing the hyperparameters optimally and individually for each different subject. In this work, we implemented and optimized a CNN and an LSTM recurrent neural network, because such models achieve the most promising performance in the literature [33]. Both networks were trained using a subset of the available data and then tested on subsequent data of the same *in silico* patient without being updated again. The networks have a sequence-to-label architecture, as the expected output is a single value corresponding to the expected blood glucose value in 30 minutes. After splitting the data into Training (70%), Validation (20%), and Test set (10%), the models were built.

The proposed CNN is a 1D-CNN, with one-dimensional kernel, consisting of two convolutional layers with ReLU activation function, each followed by a MaxPooling that cuts the parameters in half by taking, in pairs, only the largest value. To complete the model, the convolutional layers are followed by a dense layer with ReLU activation function, and an output neuron that provides the final regression. A schematic representation of the proposed CNN model is reported in Figure 1. The choice of hyperparameters was made by performing a grid search on the validation set, based on a range of parameters including values identified through preliminary tests and parameters reported in the literature [33]. The optimization was done with respect to the kernel size and the number of feature maps.

The proposed LSTM model consists of a first LSTM layer, a dense layer with ReLU activation function, and an output layer that returns the predicted CGM value. Also in this case, the model was optimized in terms of the number of neurons in the first LSTM layer and in the dense layer by investigating both parameters identified in preliminary tests and parameters reported in the literature [33]. A schematic representation of the proposed LSTM model is reported in Figure 2.

Both models take as input a (3×30) matrix of values, corresponding to the last 30 minutes of the 3 feature values. Such parameter was identified in preliminary tests, as it provides the models with enough information to capture the recent trend of the features. We found empirically that using longer monitoring periods did not improve performance. With regards to the strategy chosen to train both networks, the Stochastic Gradient Descent (SGD) optimizer is adopted, which requires a learning rate (0.0001), a momentum (0.9) and a clip Value (0.5), which is a necessary parameter to prevent the gradient explosion phenomenon in deep neural networks, improving the prediction quality. The training of both model was performed by splitting the data into mini-batches of 1400 samples (i.e., approximately one day of data) and setting the maximum number of epochs to 200. Finally, to prevent overfitting, the early stopping strategy was adopted, which stops training if the performance on the validation set does not improve within a fixed number of consecutive epochs.

Two different evaluation metrics are used to thoroughly evaluate the performance of the models. Root Mean Square Error (RMSE) is utilized to assess numerical accuracy, as it provides a numerical estimate of how close the predicted values are to the real ones. Let

us consider a prediction performed at timestamp t. Defined $P(t+PH)$ as the prediction performed at time t regarding the future glucose value $CGM(t+PH)$, and considering a time series with a total of T timestamps to be predicted, the RMSE is defined as:

$$RMSE = \sqrt{\sum_{t=1}^{T-PH} \frac{(CGM(t+PH) - P(t+PH))^2}{T-PH}} \quad (3)$$

where PH is the considered prediction horizon. The smaller the RMSE value, the better the performance. In addition, we considered the Clarke Error Grid (CEG) analysis as a measure of the clinical accuracy of the predictions produced. The CEG consists in a grid which is divided into 5 zones, from A to E, which plots the actual and the predicted CGM values on the horizontal and the vertical plot axis, respectively. Values in zones A and B represent good or acceptable glucose predictions; values in zone C represent mistaken predictions that may lead to unnecessary treatment; values in zone D represent a dangerous failure to predict; finally, values in zone E represent a completely wrong prediction that would lead to erroneous treatment [22].

2.3. Edge system description

In order to test the feasibility of the predictive models of being implemented and utilized on an edge system, we needed to identify the target hardware. Our choice fell on two different devices: a Raspberry Pi4, chosen for its low cost and high computational capability, and a Coral DevBoard, a developer kit containing a Tensor Processing Unit (TPU) processor which is useful for accelerating the execution of machine learning models. The Raspberry Pi4 has a Broadcom BCM2711 quad-core Arm Cortex A72 of 1.5GHz processor, with 4 GB of memory. Furthermore, in order to be able to carry out the tests, we chose to use Raspbian OS (a Debian-derived ISO) as operating system. Python and Mendel Development Tool (MDT) were also installed. The former is necessary to perform tests directly on the Raspberry; the latter is used to give commands to the Coral DevBoard, and therefore allows its set-up and use. The Coral Devboard has a quad Cortex-A53, Cortex-M4F CPU, with 1 GB LPDDR4 RAM, and it has a 4 TOPS (8bit) TPU accelerator for machine learning processes. The operating system running on the DevBoard is Mendel Linux. We installed and utilized all the dependencies necessary to run the model on the board using the Py CoralAPI.

2.4. Edge system implementation

Both datasets were provided as input, as sequences of the last 30 minutes of values, for two models compared: CNN and LSTM. The models were implemented and trained on Google Colab through the use of the open source libraries of Keras and TensorFlow. Through this API, the networks were trained and the hyperparameters optimized.

Although the single models were trained on two different datasets, topologically the trained networks do not differ, in terms of hyperparameters. Therefore, the number of algebraic operations performed by a single network is invariant with respect to the dataset. Having made this consideration we decided to implement on the edge device only the models trained on the dataset including more hypo/hyperglycemic events, as it is more similar to a real use case.

For the implementation of the models on edge computing architectures, it is necessary to perform a quantization step that differs depending on the architecture on which inference is going to be performed. In order to perform regression tasks on the Rasperry, we chose to use the quantization in *.tflite* format, that transforms the model keeping output variables in *float32* format. This optimization, namely dynamic range quantization, provides latency close to fully fixed-point inference. However, the outputs are still stored using floating point so that the speedup with dynamic-range operations is less than a full fixed-point computation, as reported on the official TensorFlow web page [34]. From now on we will refer to the model obtained with this quantization as *.tflite*.

For the implementation on the Dev Board, it was necessary to transform the models in their 8-bit representation, in order to execute them exploiting the full potential provided by the Coral's TPU. In this case, the quantization method to be used is known as full integer quantization. Applying this approach requires to provide a representative dataset, in order to calibrate variable tensors such as model input, activation functions, outputs of intermediate layers, and model output. As a representative dataset it would theoretically be sufficient to provide a set of 100-500 sample data, taken between the training and validation set. In our case, a dependence of the goodness of the quantization on the subset of data passed to the model as a representative dataset was noted. In fact, it was not sufficient to use data taken randomly from the training or validation set but it was necessary to use ordered data, given the time series forecasting nature of the task. At the end of this quantization procedure, all input and output values are taken to *uint8*. From now on we will refer to the model obtained with this quantization as *uint8*.

Due to the 8-bit nature of the quantization required to exploit the capabilities of the Coral Devboard TPU processor, a problem arose for our regression task. The range of values of the dataset varies between 10 and 600 mg/dl, whereas the values that can be represented with 8 bits are 256. Consequently, we pursued two approaches. The first consists in avoiding any pre-processing of the input data, and then reconstructs the possible overflow cases obtained in the output through a post-processing of the data, maintaining the granularity of the prediction at 1 mg/dl. The reconstruction was done following the procedure set out in the algorithm 1. It assumes that a decrease of glucose concentration of more than 50 mg/dl in a single minute is very unlikely or impossible. In this case, we post-process the prediction and sum 255 to the predicted value.

Algorithm 1 Output reconstruction algorithm

```
reconstructed_pred = []                                   ▷ initialization of variables
overflow = False
deltaY = 50
For i,x in enumerate (tflite_uint8_model_prediction):     ▷ Start of the for loop
if x >= 240 then
    if overflow and (x - tflite_uint8_model_prediction[i-1]) >= deltaY: then
        overflow = False
    else if  not overflow and (x - tflite_uint8_model_prediction[i+1]) >= deltaY: then
        overflow = True
delta = 255 if overflow else 0
reconstructed_pred.append(x + delta)                      ▷ End of the for loop
```

The second approach consists in the application of a normalization step in the pre-processing phase, remapping the data values between 0 and 255. Such an approach avoids problems related to overflow, but it takes the granularity of the prediction to approximately 2.33 mg/dl. Then, we de-normalized the predicted values to compute the evaluation metrics. This could introduce inaccuracy in the predictions.

The Raspberry and DevBoard were used for the calculation of inference times, to be compared with the performance limits that our application requires (less than the sampling period of the sensor, i.e. 1 minute). At each timestamp, the edge system takes as input the 30 most recent values of the features (i.e., the data of the *in silico* patient produced by the simulator), computes the latest value of the IOB, and performs a prediction of the future blood glucose level. A representative schematic of the experimental system can be seen in Figure 3.

3. Results and Discussion

As a result of the grid search performed on the Discovery set, the optimal configuration of the CNN comprises a number of filters equal to 26 for the first convolutional layer, 20 filters for the second convolutional layer, and a Kernel size equal to 1x5 on both. Note that, due to the shape chosen for the filters and to the structure of the input matrix, in the first

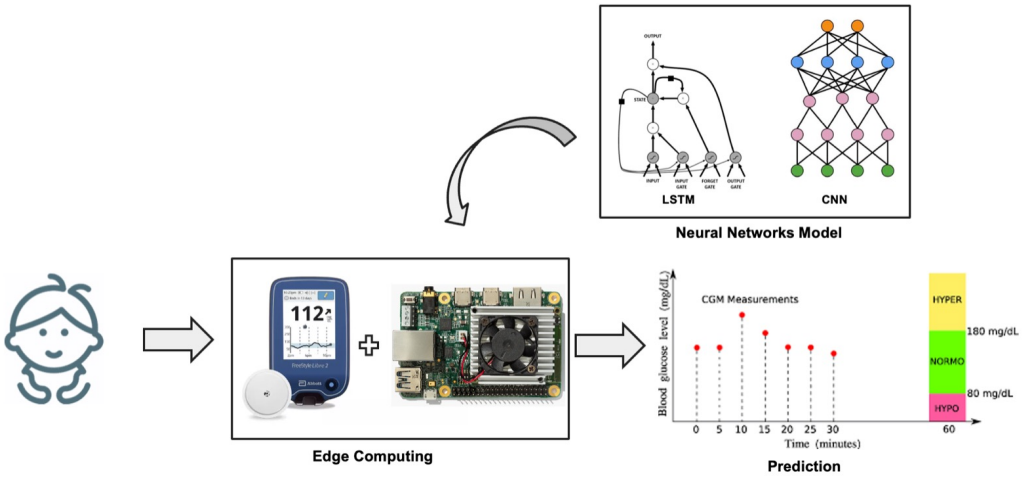

Figure 3. Schematic representation of the experimental setup during the test phase with edge systems.

Table 1. Results of the tests performed with the proposed models CNN and LSTM with out carring the normalization step in the pre-processing phase. The results refer to the RMSE [mg/dl] achieved on both the ideal (no-error) and the realistic (hypo-hyper) dataset. Such results are reported in terms of average RMSE $\pm$ standard deviation. The CEG results are referred only to the realistic dataset, and its results are reported as percentage on the total dataset. For each neural network, we reported the results for the model implemented on Google Colab, for the model implemented on Raspberry (*.tflite float32* format), and for the model implemented on the Dev Board (.tflite *uint8*).

Model	RMSE (no-error)	RMSE (hypo-hyper)	CEG (A;B;C;D;E)
CNN	22.2 ± 2.5	23.2 ± 2.3	87.0; 12.0; 0.0; 1.0; 0.0
LSTM	13.5 ± 3.4	16.3 ± 4.7	93.8; 5.2; 0.0; 1.0; 0.0
CNN *.tflite*	/	23.6 ± 2.0	85.7; 13.6; 0.0; 0.7; 0.0
LSTM *.tflite*	/	16.3 ± 4.7	93.7; 5.2; 0.0; 1.1; 0.0
CNN *uint8*	/	40.1 ± 11.1	75.4; 20.8; 0.0; 1.2; 2.5
LSTM *uint8*	/	35.0 ± 13.3	82.4; 12.5; 0.0; 1.5; 3.6

CNN layer the convolutions are performed on different timestamps of the same feature. With regards to the LSTM model, the optimal configuration resulted in 64 neurons for both the LSTM and the fully-connected layer. Once the models were optimized, predictions were performed on the Test set, and the RMSE and the CEG were computed. With regards to the CEG values, only those from the second dataset were evaluated, as they present more hypo- and hyperglycemic values and are thus more similar to a real-life scenario.

Table 1 reports the average values and their standard deviation of the tests performed using the different versions of the models. As expected, the results achieved by the baseline model on the standard dataset are better than those achieved on the dataset with outliers. The LSTM model outperforms the CNN on both datasets, both in terms of average RMSE and CEG results. In particular, with regards to the realistic dataset, the LSTM achieves an RMSE of $16.3 \pm 4.7\, mg/dl$, which is noteworthy if compared to other studies presented in the literature concerning the prediction on pediatric T1D patients. Also, 99.0% of its predictions fall in zones A and B of the CEG and thus represent clinically accurate or acceptable predictions, whereas 1.0% of predictions fall in zone D. The latter mainly correspond to failures of predicting hypoglycemia. No predictions fall in zones C and E. The following sections analyze the performance of the models after the implementation on the edge devices.

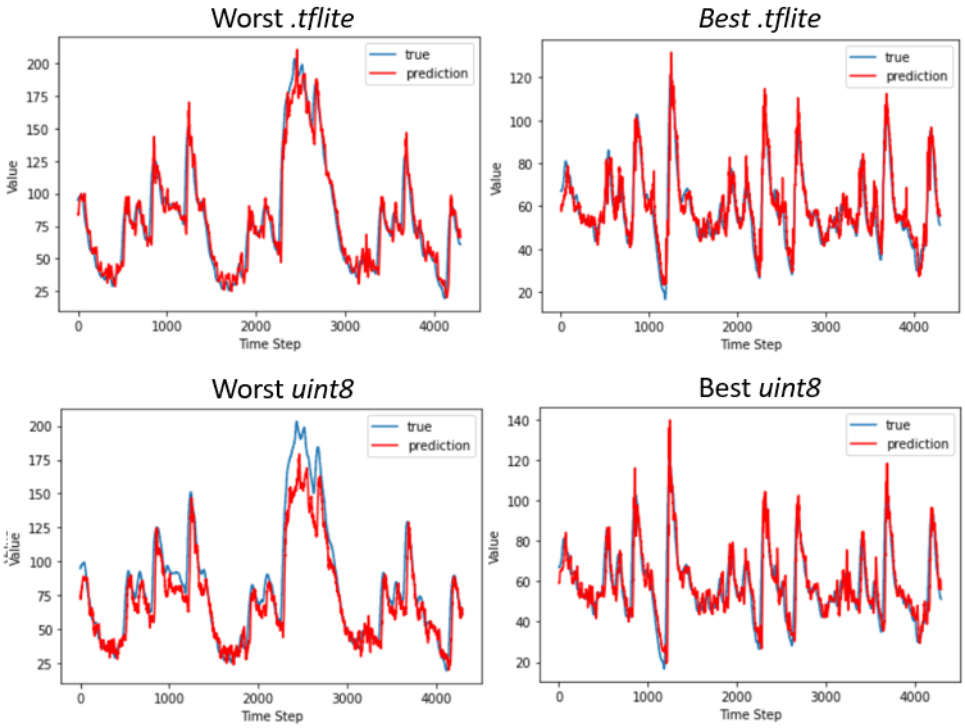

Figure 4. Graphical examples of the best and worst predictions performed by the CNN (left) and LSTM (right) using different edge devices. The glycemic index value shown in the figure is normalised between 0 and 255, so to obtain the real glycemic value we need to multiply by 2,33.

3.1. Edge system results and discussions

The results reported in Table 1 refer to the models trained without having carried out the normalization phase of the input values. The expected increase in the RMSE values of the models implemented on the edge devices can be observed; however, this variation differs between the two quantized representations of the networks. With regards to models quantized using dynamic range quantization for implementation on the Raspberry, the RMSE values increase by a maximum of 0.4 mg/dl for the CNN, whereas there is no difference for the LSTM. Again, the LSTM model outperforms the CNN in terms of numerical accuracy, achieving an RMSE of 16 ± 4.7 mg/dl, and 98.9% of its predictions fall in zones A and B of the CEG. This result is of particular interest because it is similar to the performance achieved on datasets composed of data of adult T1D patients, and it is achieved on the edge device, without resorting to cloud computing. A graphical example of the predictions is reported in Figure 4, where we report as an example data of two patients for whom the best and the worst performance is achieved in terms of RMSE. The LSTM prediction is closer to the true CGM value compared to the CNN, which produces more oscillatory predictions; however, the LSTM tends to overestimate both hyperglycemic and hypoglycemic peaks.

Nonetheless, it is worth noting that only 0.7% of predictions of the CNN model fall outside the A and B zones of the CEG, compared to 1.1% of the LSTM; conversely, the LSTM produces more predictions that fall in zone A (93.7% against 85.7% of the CNN). This may be explained considering that the LSTM is more capable of performing accurate predictions in the euglycemic range, which translates into better RMSE and a larger percentage of predictions in zone A, whereas it may miss some hypoglycemic events; on the contrary, the CNN has a larger RMSE and a larger amount of predictions in zone B of the CEG, corresponding to errors in the euglycemic range, whereas it is more capable to predict hypoglycemia. Examples of the CEG are shown in Figure 5, where we report as an example data of two patients for whom the best and the worst performance is achieved in terms of CEG percentage in zone A. In conclusion, the CNN may be more appropriate to predict

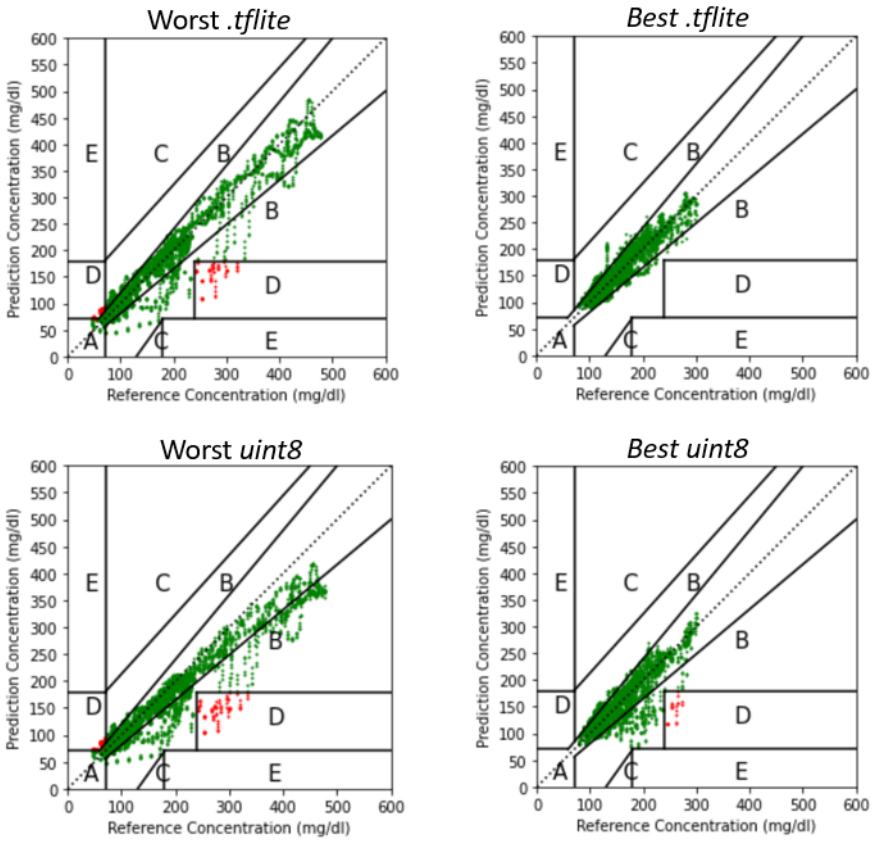

Figure 5. Clarke Error Grids resulted by the best and worst predictions of the CNN (left) and LSTM (right) using different edge devices. Predictions falling in the safe zones A and B are plotted in green; predictions in zone C are plotted in yellow; predictions falling in the dangerous zones D and E are plotted in red.

critical hypoglycemic events when implemented in *.tflite*, although its average numeric accuracy is worse than that of LSTM. However, it should be taken into account that results achieved on virtual patients are, in general, slightly better than those obtained on real patients, thus performance may deteriorate when testing on a real dataset.

A different analysis applies to the models on which the full integer quantization was performed for implementation on the Coral DevBoard. Indeed, this quantization technique, that casts the values from *float32* to *uint8*, has more significant effects on the goodness of prediction. In particular, the overflow that is observed when glycemic values are above 255 mg/dl considerably increases the RMSE scores, and generates some predictions that fall in the dangerous E zone of the CEG. For this reason, as explained in section 2.4, two different approaches were chosen. The second one, which involved an initial pre-processing of the data, gave considerably better results than the first one, and they are reported in Table 2. In particular, the results obtained for the models in Google Colab do not differ substantially from those achieved without the normalization; conversely, the *uint8* implementation of such models achieves considerably better performance than those obtained with the first approach. It must be considered that the granularity of the prediction increases from 1 mg/dl to 2.3 mg/dl. In spite of this drawback, we can still consider this approach better than the first one, because the increase in granularity obtained is not critical from a clinical point of view. It is worth noting that, although the LSTM model outperforms the CNN in terms of RMSE (21.2 ± 8.6 and $24.7 \pm 5.5\, mg/dl$, respectively), 5% of the predictions produced by the LSTM fall in the D zone of the CEG, corresponding to a failure of predicting dangerous events. This situation shows the LSTM model to be weaker to the *uint8* representation, which brings it a greater drop in accuracy. This is probably due to the narrowness of

Table 2. Results of the tests performed with the proposed models CNN and LSTM, on which was carried the normalization step in the pre-processing phase. The results refer to the RMSE [mg/dl] achieved on the realistic (hypo-hyper) dataset. Such results are reported in terms of average RMSE $\pm$ standard deviation. The CEG results are referred only to the realistic dataset, and its results are reported as percentage on the total dataset. For each neural network, we reported the results for the model implemented on Google Colab, and for the model implemented on the Dev Board (.tflite uint8 format).

Model	RMSE (hypo-hyper)	CEG (A;B;C;D;E)
CNN	21.8 ± 2.3	87.8; 10.9; 0.0; 1.1; 0.0
LSTM	16.0 ± 3.4	93.7; 5.5; 0.0; 0.8; 0.0
CNN *uint8*-normalized	24.7 ± 5.5	87.6; 9.8; 0.0; 0.9; 0.0
LSTM *uint8*-normalized	21.2 ± 8.6	87.4; 7.5; 0.0; 5.1; 0.0

Table 3. Maximum inference time obtained in the test phase in milliseconds. The inference times are reported for each model, CNN and LSTM. They were calculated: for the models saved in TensorFlow saved model format over the Colab online TPU, for the *.tflite* model format over the Raspberry and for the *.tflite* format quantizated in *uint8* over the Coral DevBoard.

Model	Colab TPU (TF Saved Model)	Raspberry (*.tflite*)	Coral DevBoard (*.tflite uint8*)
CNN	0.085	101.56	18
LSTM	0.086	70.3	12

the model, which has only one LSTM plane. Given the limited number of mathematical operations required to achieve an output, the conversion step of the model to *uint8* fails to optimize the weights with the new integer values. On the contrary, only 0.9% of the predictions produced by the CNN fall in the D zone, proving that this latter model is more clinically accurate and reliable when implementing the models in *uint8*, despite the better numerical accuracy achieved by the LSTM model.

A further comparison between the different implementation concerns the actual inference times obtained, which returned largely satisfying results. We reported in Table 3 the worst-case results for each model and hardware to show compliance with the time constraints posed by the application. The inference times for both models in all three representations are far below the limit imposed by the application, i.e. 1 minute. However, the total times in the case of a real application should also take into account the times necessary for: signal collection by the sensors, pre-processing of the raw data, and displaying the results on an appropriate Graphic User Interface (GUI). Nonetheless, the time for a single inference operation to be summed are, in the worst case, the ones of the CNN performed in *.tflite* format by the Raspberry, corresponding to 101.56 *ms*. We can therefore assert that inference times, covering at most 0.17% of the total time limit imposed by the application, are not one of the parameters to be optimized in the case of a real implementation of the system. Furthermore, looking at Table 3 and comparing the data obtained in the tests of the two Edge systems, a consistent acceleration can be observed with the use of the Coral DevBoard when compared to the Raspberry's performance, although it does not reach the performance of Google Colab TPU. This result is in line with Google's own claims [35].

4. Conclusions

In this manuscript, we implemented a CNN and an LSTM neural network for the prediction of blood glucose concentration in pediatric T1D patients. The UVA/Padova simulator was exploited to generate data of 10 virtual children, and 2 datasets were generated which differ for the amount of hypoglycemic and hyperglycemic events. We determined the optimal parameters of the models through the medium of a grid search on the Discovery set, and evaluated performance by the predictions on the Test set using Google Colab, a Rasp-

berry, and a Coral DevBoard. To the best of our knowledge, this is the first attempt to implement an edge-computing system for the prediction of glucose concentration in children.

With regards to the prediction of glucose levels, the models achieved numerical accuracy comparable to those reported in the literature for adult patients. However, we acknowledge that, since the results are achieved on virtual patients, they may not be fully representative of the actual predictive capabilities of the models. On the one hand, the LSTM model achieved the best numerical accuracy and the largest percentage of predictions in zone A of the CEG for all the tests performed without model quantization. On the other hand, the CNN model produced a smaller percentage of predictions in the dangerous zones of the CEG with respect to all the implementations on edge devices, proving to be more effective in predicting critical events. In conclusion, both proposed models are promising for a possible real implementation on pediatric patients.

With regards to the edge computing, we arrived at a double result. On the one hand, the loss of information and prediction quality was tested with respect to two different quantizations of the networks. Both approaches achieved results comparable to those achieved using Google Colab. The *.tflite* implementation achieved the best results, although the *uint8* showed smaller inference times. On the other hand, the tests on inference times showed us that the IoT devices currently on the market have sufficient computational capabilities to be used in applications that require time constraints such as the one imposed by our specific case study, i.e. 1 minute. In conclusion, the *.tflite* implementation seems more promising, because it achieves the best results and there is no particular concern about the inference time.

Several future developments may follow this work. First, it would be interesting to validate the proposed neural networks on data of real patients, in order to confirm the good performance achieved on virtual patients. Second, a mobile application could be developed to provide the patient with real-time information about their future glycemic value, and generate an alarm in case of dangerous conditions by directly interacting with the edge device. Such application may also collect a history of the patient's data in order to allow the physicians to adjust the therapy. Finally, it would be interesting to develop a complete proof of concept including also the acquisition system in order to exploit its actual limits and potential.

Author Contributions: **Federico D'Antoni:** Conceptualization, Methodology, Formal analysis, Validation, Writing - original draft. **Lorenzo Petrosino:** Methodology, Software, Investigation, Writing - original draft, Visualization. **Fabiola Sgarro:** Software, Formal analysis, Data curation, Visualization.**Antonio Pagano:** Software, Data Curation **Luca Vollero:** Methodology, Writing - review and editing, Supervision. **Vincenzo Piemonte:** Conceptualization, Writing - review and editing **Mario Merone:** Conceptualization, Methodology, Validation, Writing - original draft, Supervision, Project administration. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by POR FESR Lazio 2014-2020. Progetto T0002E0001. "Progetti di Gruppi di Ricerca 2020" [grant number: A0375E0196]

.

Data Availability Statement: Restrictions apply to the availability of these data. Data was obtained from The Epsilon Group and are available at https://tegvirginia.com/software/t1dms-2014/ with the permission of The Epsilon Group.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Sample Availability: Samples of the scenario generation codes are available from the authors.

References

1. Reid, L.; Baxter, F.; Forbes, S. Effects of islet transplantation on microvascular and macrovascular complications in type 1 diabetes. *Diabetic Medicine* **2021**, *38*, e14570. doi:10.1111/dme.14570.

2. Polonsky, W.H.; Hessler, D.; Ruedy, K.J.; Beck, R.W. The impact of continuous glucose monitoring on markers of quality of life in adults with type 1 diabetes: further findings from the DIAMOND randomized clinical trial. *Diabetes Care* **2017**, *40*, 736–741. doi:10.2337/dc17-0133.
3. Toschi, E.; Michals, A.; Adam, A.; Davis, D.; Atakov-Castillo, A.; Slyne, C.; Munshi, M. Usefulness of CGM-Derived Metric, the Glucose Management Indicator, to Assess Glycemic Control in Non-White Individuals With Diabetes. *Diabetes Care* **2021**, *44*, 2787–2789. doi:10.2337/dc21-1373.
4. Woldaregay, A.Z.; Årsand, E.; Botsis, T.; Albers, D.; Mamykina, L.; Hartvigsen, G. Data-driven blood glucose pattern classification and anomalies detection: machine-learning applications in type 1 diabetes. *Journal of medical Internet research* **2019**, *21*, e11030. doi:10.2196/11030.
5. Piemonte, V.; Capocelli, M.; De Santis, L.; Maurizi, A.R.; Pozzilli, P. A novel three-compartmental model for artificial pancreas: development and validation. *Artificial organs* **2017**, *41*, E326–E336. doi:10.1111/aor.12980.
6. Lynch, S.M.; Bequette, B.W. Estimation-based model predictive control of blood glucose in type I diabetics: a simulation study. Proceedings of the IEEE 27th Annual Northeast Bioengineering Conference (Cat. No. 01CH37201). IEEE, 2001, pp. 79–80. doi:10.1109/ACC.2002.1024561.
7. Oviedo, S.; Vehí, J.; Calm, R.; Armengol, J. A review of personalized blood glucose prediction strategies for T1DM patients. *International journal for numerical methods in biomedical engineering* **2017**, *33*, e2833. doi:10.1002/cnm.2833.
8. De Paoli, B.; D'Antoni, F.; Merone, M.; Pieralice, S.; Piemonte, V.; Pozzilli, P. Blood Glucose Level Forecasting on Type-1-Diabetes Subjects during Physical Activity: A Comparative Analysis of Different Learning Techniques. *Bioengineering* **2021**, *8*, 72. doi:10.3390/bioengineering8060072.
9. Bunescu, R.; Struble, N.; Marling, C.; Shubrook, J.; Schwartz, F. Blood glucose level prediction using physiological models and support vector regression. 12th Int. Conf. on Machine Learning and Applications. IEEE, 2013, Vol. 1, pp. 135–140. doi:10.1109/ICMLA.2013.30.
10. Georga, E.I.; Protopappas, V.C.; Polyzos, D.; Fotiadis, D.I. A predictive model of subcutaneous glucose concentration in type 1 diabetes based on random forests. Int. Conf. of the IEEE Eng. in Medicine and Biology Society, 2012, pp. 2889–2892. doi:10.1109/EMBC.2012.6346567.
11. Sparacino, G.; Zanderigo, F.; Corazza, S.; Maran, A.; Facchinetti, A.; Cobelli, C. Glucose concentration can be predicted ahead in time from continuous glucose monitoring sensor time-series. *IEEE Transactions on Biomedical Engineering* **2007**, *54*, 931–937. doi:10.1109/TBME.2006.889774.
12. D'Antoni, F.; Merone, M.; Piemonte, V.; Iannello, G.; Soda, P. Auto-Regressive Time Delayed jump neural network for blood glucose levels forecasting. *Knowledge-Based Systems* **2020**, p. 106134. doi:10.1016/j.knosys.2020.106134.
13. Li, K.; Liu, C.; Zhu, T.; Herrero, P.; Georgiou, P. GluNet: A Deep Learning Framework For Accurate Glucose Forecasting. *IEEE journal of biomedical and health informatics, 2019* **2019**. doi:10.1109/JBHI.2019.2931842.
14. Mosquera-Lopez, C.; Dodier, R.; Tyler, N.; Resalat, N.; Jacobs, P. Leveraging a big dataset to develop a recurrent neural network to predict adverse glycemic events in type 1 diabetes. *IEEE Journal of Biomedical and Health Informatics* **2019**. doi:10.1109/JBHI.2019.2911701.
15. Li, K.; Daniels, J.; Liu, C.; Herrero, P.; Georgiou, P. Convolutional recurrent neural networks for glucose prediction. *IEEE journal of biomedical and health informatics* **2019**, *24*, 603–613. doi:10.1109/JBHI.2019.2908488.
16. Visentin, R.; Campos-Náñez, E.; Schiavon, M.; Lv, D.; Vettoretti, M.; Breton, M.; Kovatchev, B.P.; Dalla Man, C.; Cobelli, C. The UVA/Padova type 1 diabetes simulator goes from single meal to single day. *Journal of diabetes science and technology* **2018**, *12*, 273–281. doi:10.1177/1932296818757747.
17. Danne, T.; Battelino, T.; Jarosz-Chobot, P.; Kordonouri, O.; Pánkowska, E.; Ludvigsson, J.; Schober, E.; Kaprio, E.; Saukkonen, T.; Nicolino, M.; et al. Establishing glycaemic control with continuous subcutaneous insulin infusion in children and adolescents with type 1 diabetes: experience of the PedPump Study in 17 countries. *Diabetologia* **2008**, *51*, 1594–1601. doi:10.1007/s00125-008-1072-2.
18. Worth, C.; Dunne, M.; Ghosh, A.; Harper, S.; Banerjee, I. Continuous glucose monitoring for hypoglycaemia in children: perspectives in 2020. *Pediatric Diabetes* **2020**, *21*, 697–706. doi:10.1111/pedi.13029.
19. Mougiakakou, S.G.; Prountzou, A.; Iliopoulou, D.; Nikita, K.S.; Vazeou, A.; Bartsocas, C.S. Neural network based glucose-insulin metabolism models for children with type 1 diabetes. 2006 International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE, 2006, pp. 3545–3548. doi:10.1109/IEMBS.2006.260640.
20. Dassau, E.; Cameron, F.; Lee, H.; Bequette, B.W.; Zisser, H.; Jovanovič, L.; Chase, H.P.; Wilson, D.M.; Buckingham, B.A.; Doyle III, F.J. Real-time hypoglycemia prediction suite using continuous glucose monitoring: a safety net for the artificial pancreas. *Diabetes care* **2010**, *33*, 1249–1254. doi:10.2337/dc09-1487.
21. De Bois, M.; El Yacoubi, M.A.; Ammi, M. Study of short-term personalized glucose predictive models on type-1 diabetic children. 2019 International Joint Conference on Neural Networks (IJCNN). IEEE, 2019, pp. 1–8. doi:10.1109/IJCNN.2019.8852399.
22. Clarke, W.L. The original Clarke error grid analysis (EGA). *Diabetes technology & therapeutics* **2005**, *7*, 776–779. doi:10.1089/dia.2005.7.776.
23. Nasser, A.R.; Hasan, A.M.; Humaidi, A.J.; Alkhayyat, A.; Alzubaidi, L.; Fadhel, M.A.; Santamaría, J.; Duan, Y. IoT and Cloud Computing in Health-Care: A New Wearable Device and Cloud-Based Deep Learning Algorithm for Monitoring of Diabetes. *Electronics* **2021**, *10*, 2719. doi:10.3390/electronics10212719.

24. Bhat, G.M.; Bhat, N.G. A novel IoT based framework for blood glucose examination. 2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT). IEEE, 2017, pp. 205–207. doi:10.1109/ICEECCOT.2017.8284666.
25. Li, E.; Zeng, L.; Zhou, Z.; Chen, X. Edge AI: On-Demand Accelerating Deep Neural Network Inference via Edge Computing. *IEEE Transactions on Wireless Communications* **2020**, *19*, 447–457. doi:10.1109/TWC.2019.2946140.
26. Zhou, Z.; Chen, X.; Li, E.; Zeng, L.; Luo, K.; Zhang, J. Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing. *Proceedings of the IEEE* **2019**, *107*, 1738–1762. doi:10.1109/JPROC.2019.2918951.
27. Hersh, W.R.; Helfand, M.; Wallace, J.; Kraemer, D.; Patterson, P.; Shapiro, S.; Greenlick, M. Clinical outcomes resulting from telemedicine interventions: a systematic review. *BMC Medical Informatics and Decision Making* **2001**, *1*, 1–8. doi:10.1186/1472-6947-1-5.
28. Scott Kruse, C.; Karem, P.; Shifflett, K.; Vegi, L.; Ravi, K.; Brooks, M. Evaluating barriers to adopting telemedicine worldwide: a systematic review. *Journal of telemedicine and telecare* **2018**, *24*, 4–12. doi:10.1177/1357633X16674087.
29. Zhu, T.; Kuang, L.; Daniels, J.; Herrero, P.; Li, K.; Georgiou, P. IoMT-Enabled Real-time Blood Glucose Prediction with Deep Learning and Edge Computing. *IEEE Internet of Things Journal* **2022**. doi:10.1109/JIOT.2022.3143375.
30. Dungan, K.M.; Sagrilla, C.; Abdel-Rasoul, M.; Osei, K. Prandial insulin dosing using the carbohydrate counting technique in hospitalized patients with type 2 diabetes. *Diabetes Care* **2013**, *36*, 3476–3482. doi:10.2337/dc13-0121.
31. Zisser, H.; Robinson, L.; Bevier, W.; Dassau, E.; Ellingsen, C.; Doyle III, F.J.; Jovanovic, L. Bolus calculator: a review of four "smart" insulin pumps. *Diabetes technology & therapeutics* **2008**, *10*, 441–444. doi:10.1089/dia.2007.0284.
32. Toffanin, C.; Zisser, H.; Doyle III, F.J.; Dassau, E. Dynamic insulin on board: incorporation of circadian insulin sensitivity variation. *Journal of diabetes science and technology* **2013**, *7*, 928–940. doi:10.1177/193229681300700415.
33. Idrissi, T.E.; Idri, A. Deep Learning for Blood Glucose Prediction: CNN vs LSTM. International Conference on Computational Science and Its Applications. Springer, 2020, pp. 379–393. doi:10.1007/978-3-030-58802-1_28.
34. TensorFlow. Post-training quantization. *https://www.tensorflow.org/lite/performance/post_training_quantization* **2022**.
35. Google. Frequently asked questions. *available online at: https://coral.ai/docs/edgetpu/faq/#how-is-the-edge-tpu-different-from-cloud-tpus* **2022**.