A Cloud-Hosted Online Learning Approach for Glycemic Index Forecasting *

Lazzarinetti Giorgio[1] [0000−0003−0326−8742], Massarenti Nicola[1] [0000−0002−8882−4252], and Mantisi Stefania[1] [0000−0003−4446−9743]

Noovle S.p.A, Milan, Italy https://www.noovle.com/en/

Abstract. Recent developments in diabetes care technologies, like wearable devices for continuous glucose monitoring, have made it possible for patients to access relevant data and properly treat diabetes mellitus diseases. Thanks to machine learning, moreover, it has been possible to predict the trend of the level of glucose in the blood in real-time, thus preventing diseases. In this research, we propose to compare the two main state-of-the-art algorithms for time series forecasting, namely Autoregressive Integrated Moving Average (ARIMA) models and Recurrent Neural Network (RNN) models. In particular, we propose an auto adaptive algorithm for ARIMA models' parameters selection based on the Augmented Dickey-Fuller test for stationarity and autocorrelation and partial autocorrelation functions for autoregressive and moving average orders and we show that somehow this would make the ARIMA model preferable to RNN-based models in an online learning scenario for time series forecasting, with a root mean square error of 1.11 mmol/l. We also propose some considerations related to a Google Cloud based infrastructure to host an online learning application, comparing the performance of the tested models also with respect to their training time and their scalability and maintenance, showing that ARIMA models clearly require a lighter infrastructure and a less complex pipeline for managing model life cycle with respect to RNN-based models.

Keywords: Continuous Glucose Monitoring · Time Series Forecasting · Autoregressive Integrated Moving Average · Recurrent Neural Network · Online Learning.

1 Overview

Diabetes mellitus is a metabolic disorder that causes blood glucose level to deviate from normal values and can lead serious health complications and in some cases, if not properly treated, even death [1]. Currently, there is no cure for diabetes, however keeping

* Activities were partially funded by Italian "Ministero dello Sviluppo Economico", Fondo per la Crescita Sostenibile, Bando "Agenda Digitale", D.M. Oct. 15th, 2014 - Project n. F/020012/02/X27 - "Smart District 4.0".

blood glucose levels within the recommended range is a key factor in treating this discomfort. This includes monitoring blood glucose levels, exercising, observing a strict diet and following an insulin-based treatment [2, 3]. Recent developments in diabetes care technologies have made it easier for patients to access relevant data. In particular, there are continuous glucose monitoring (CGM) technologies that record the level of glucose in the blood at intervals of a few minutes. CGM technologies have the potential to be used for the prediction of blood glucose concentration with the consequent optimization of glycemic control. Furthermore, recently, machine learning and data mining techniques have reached a sufficient degree of maturity in forecasting problems to allow for the prediction of the glycemic index trend. In this research, we present a comparison of the state-of-the-art algorithms for glucose level forecasting with the aim of developing an online learning framework, which is able to continuously adapt to changes. In particular, we compare Autoregressive Integrated Moving Average (ARIMA) models with Recurrent Neural Network (RNN) models and we test their ability to predict the glucose level's trend in the near future (30 to 60 minutes). Moreover, we developed an auto-adaptive learning algorithm to optimize the ARIMA parameter in an online learning fashion. The approach for hyperparameter optimization we defined uses the Augmented Dikey-Fuller (ADF) test for stationarity condition to define the differentiation order and the autocorrelation (ACF) and partial autocorrelation functions (PACF) to select the range for the autoroegressive and moving average parameters, then optimized with the Akaike's Information Criterion (AIC). Differently from other researches in the field, our algorithm does not perform optimization on a predefined range of the autoregressive and moving-average steps, but select these ranges considering the trends and the statistical significance of each point of the ACF and PACF. The following is organized as follows. In Chapter 2 the most used approaches for the prediction of time series will be described with a focus on the prediction of the glycemic index. In Chapter 3 we describe the dataset used for training the model and we furtherly explain the business need behind this research. In Chapter 4 we describe in detail the approaches that we intend to test for the objective and in Chapter 5 we present the experimental results. Finally, in Chapter 6 we describe the implemented architecture for the online learning purpose and in Chapter 7 some conclusive remarks.

2 State of the art

In descriptive statistics, a time series is defined as a collection of constant points in time and expresses the dynamics of a certain phenomenon over time. The analysis of time series has two main practical applications: on the one hand, to provide an interpretation of a phenomenon, identifying components of trends, cyclicality and/or seasonality; on the other, to predict the future trend of the phenomenon [4]. Time series can be of two types: deterministic, if the values of the variable can be exactly determined starting from the previous values without any error, stochastic,if the values can be determined only partially, thus introducing an error in the prediction. Most of the time series of real data belong to this second set and therefore require technical specification to infer the next values by introducing the lowest possible error [5]. In recent years, to overcome the issue related to diabetes mellitus, CGM technologies that record the level of glucose in the blood at intervals of a few minutes have spread. The data collected by those instruments is represented by a time series of the glycemic index. Thanks to the availability of such data, recently many studies related to glycemic index prediction have emerged. In [6–8] a taxonomy of the different types of existing algorithms for glycemic index prediction is provided. In particular, four main approaches are identified on the basis of the type of model used: physiological models [9], data-driven machine learning models [10–12], hybrid models [13] and control models [14–16]. Another distinction that can be made between the types of existing algorithms is based on the type of data that are considered as inputs. In fact, most of the studies are based on the use of CGM data as the only source of information to predict glucose levels, while others also use different inputs such as the meals consumed by patients, the doses of insulin taken and, very rarely, physical exercise performed [17, 18]. Another note on the possible distinctions of algorithms concerns the time horizon of the prediction: most algorithms focus on a short-term prediction (i.e., less than 60 minutes) [19], while others focus on a longer time horizon [20, 21]. Among these techniques, the most used are certainly data-driven machine learning models. In a systematic study of models based on neural networks [8] which is dated back to 2019, it is shown how most of the studies, 20% of those analyzed, privilege feed-forward neural networks (FFNN) as a prediction tool, 18% RNN in various form and 19% an hybridization of physiology-based model and machine learning. As explained in [22], the ability of RNNs to model time sequences is still an extensively studied topic that has shown interesting results. In [26], for example, Martinsson et al. propose a method based on RNNs trained using only the historical glucose level. Their approach, evaluated in terms of root mean square error (RMSE), obtains results comparable with approaches proposed by other researchers. In [27], on the other hand, the objective of the research is to compare the performance of an approach based on RNN with respect to one based on FFNN both on short-term and long-term predictions. The result of this research is that RNNs outperform FFNNs on long-term predictions, while performance is similar on medium- and short-term predictions. A different approach, but always based on RNN, is presented in [12]. Here, the model that is used is a recurrent convolutional network. The objective of the study is to evaluate the effectiveness of the prediction in the short- and medium-term, comparing the performances obtained in the training phase on a synthetic dataset and the performances obtained in the prediction phase on a real dataset. Although the performances on the real dataset are worse than those reported on the synthetic dataset, the results are still competitive with respect to the state of the art when compared. Besides deep learning techniques, more classical, but still very performing approaches in the field of time series are represented by the statistical ARIMA models. The power of these models lies in their ability to model the time series while keeping the degree of complexity of the model low. ARIMA models combine two prediction methods: an autoregressive (AR) method and a moving average (MA) method which represents the baseline statistical models for time series forecasting. Even though there are also simpler models, as AR, MA, ARMA or ARMAX, there are several studies that show how ARIMA models can be successfully applied to glucose level forecasting [23–25]. For example, in [23] a study on the application of ARIMA models to a case of prediction of the blood glucose level is presented. The proposed methodology is based on the identification of the model definition parameters in an optimal way with respect to the forecasting task and subsequently on their application to the real case. The model definition parameters are the auto-regression order (i.e. the number of previous measurements needed to determine the current value), the moving average order (i.e. the number of previous white noise values needed to determine the current value) and the degree of differentiation (i.e. the number of differentiations necessary to make the series stationary). In [25], an ARIMA-based model is proposed that uses Akaike's information verification test to determine the value of parameters in an adaptive way. This type of approach is certainly better than others because it is able to take into account the variation of the state of the series over time and is able to apply ARIMA in an adaptive way based on the trend of the series.

3 Problem Setting

The goal of this research is that of defining an online learning algorithm for glycemic index forecasting. The research is, indeed, driven by business needs since the developed algorithm is intended to be used within a system that aims at monitoring some vital parameters of bus drivers during their working hours in order to prevent diseases. In particular, data acquisition takes place thanks to different devices that monitor (with different frequencies) the blood glucose level in terms of glycemic index, the lipid profile and HbA1c, the blood pressure and heart rate. Some of these variables are acquired manually by a specific device once or twice a day. For these variables a prediction is not needed, but the collected values are visualized in order to monitor their trend. As far as the glycemic index is concerned, indeed, it is collected by a wearable device called Glunovo CGM every 3 minutes, and for this variable the goal is to predict the trend of the series in the near future, in order to alert the driver in case the trend is going out of the predefined safe boundaries. Thus, the final system must include: a component for the prediction of the glycemic index signal in the near future; an alerting system capable of reporting dangerous situations; a dashboard that can be consulted by the doctor who, based on the predictions made and the other values collected, decides whether to alert the driver or not. In this research we implement an algorithm that, at each new value registered by the Glunovo CGM device, analyzes the historical series up to that instant and provides a real-time forecast of the trend of the glycemic index for the following period with an horizon of 30 to 60 minutes. Moreover, the algorithm must be able to adapt to new registered values in an online learning fashion and it must be suitable for a huge number of drivers.

3.1 Dataset Description

In order to design the online learning framework, we use an open source dataset called D1NAMO suitable for the analyzed case. The dataset is described by F. Dubosson et al. in [29] and contains real readings of the glycemic index acquired under normal conditions using the wearable device Zephir Bioarness 3. The values of the glycemic index are acquired every 5 minutes for 9 patients with diabetes, while 6 times a day (non regularly) for 20 non-diabetic patients. The dataset also contains other information not relevant for this research since not available in the real scenario. Since for 2 out of the 9 diabetic patients the data collected were relatively scarce, the research was carried out considering just 7 distinct patients with a minimum number of 928 samples (around 3 days of data collection) for the patient with less detections up to a maximum number of 1438 for the patient with more detections (around 5 days of data collection). In Figure 1 the glucose trend for different patients is shown.

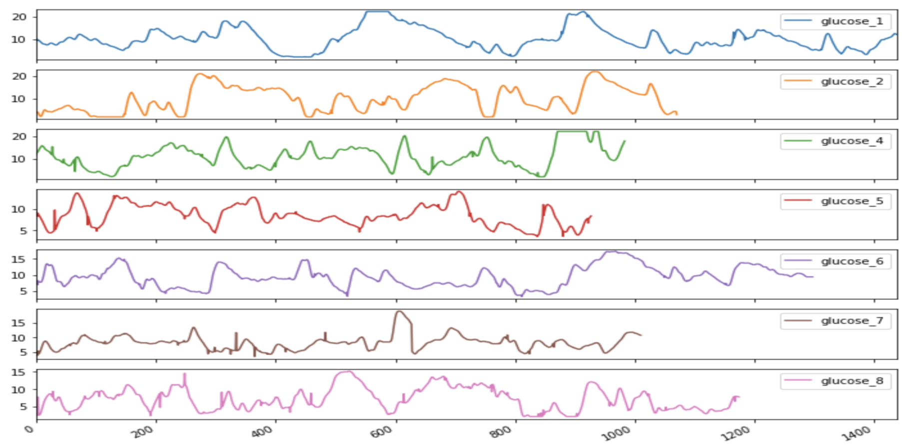

Fig. 1. Glucose trend for different patients.

4 Methodology

In designing the algorithms for glycemic index prediction, we decided to test and compare two different models that also respond to two different situations: the first (ARIMA) aims at having a model for each patient trained for each new data collected by the device in an online learning scenario; the second (RNN) aims at creating a model for each patient trained periodically in an offline model retraining scenario

4.1 ARIMA model

ARIMA models are statistical models that describe the behavior of a series considering three parameters (described in the following). Consider a time series as a variable y_t that at each instant t is described by the signal value at that instant η_t, plus an error value ϵ_t. For what follows, compare R. Adhikari et al. in [4].

$$y_t = \eta_t + \epsilon_t \tag{1}$$

A series is called autoregressive of order p, $AR(p)$, if the value of the series at instant t is a linear combination of the values of the signal from instants $t-1$ to t_p.

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \epsilon_t \quad (2)$$

A series is said to have a moving average of order q, $MA(q)$, if the value of the series at instant t is a linear combination of the values that describe the trend of the series from instants $t-1$ to t_q.

$$y_t = \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_q \epsilon_{t-q} \quad (3)$$

A series is called autoregressive moving average of orders p, q, $ARMA(p,q)$ if it is both autoregressive of order p and moving average of order q.

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_q \epsilon_{t-q} \quad (4)$$

$AR(p), MA(q)$ or $ARMA(p,q)$ models can only model stationary time series. If the joint probability distribution function of $\{y_{t-s}, ..., y_t, ..., y_{t+s}\}$ is independent of t for all s, a series y_t is strongly stationary. Thus, for a strong stationary process the joint distribution of any possible set of random variables from the process is independent of time. However for practical applications, the assumption of strong stationarity is not always needed and so a somewhat weaker form is considered. A stochastic process is said to be weakly stationary of order k if the statistical moments of the process up to that order depend only on time differences and not upon the time of occurrences of the data being used to estimate the moments. A homogeneous non-stationary series can, however, be reduced to a stationary series by introducing an appropriate degree of differentiation d. Using this degree of differentiation d, if the series is also autoregressive with moving average, the series is called autoregressive integrated with moving average of order p, d, q $ARIMA(p,d,q)$. It is important to note that neither strong nor weak stationarity implies the other. However, a weakly stationary process following normal distribution is also strongly stationary. To verify if a series is stationary it is possible to use some common statistical test like the one given by Dickey and Fuller. To determine the optimal parameters for an ARMA model, it is necessary to carry out the ACF and PACF analysis. These statistical measures reflect how the observations in a time series are related to each other. The ACF is used to measure the correlation between the current observation and an observation at lag k (using the Pearson coefficient of correlation by assuming that each variable is distributed as a Gaussian), while the PACF is used to measure the correlation between the current observation and an observation at lag k, after removing the effect of any correlation due to observations at intermediate lags (i.e. at lags < k). For each value of the ACF and PACF also a confidence interval is computed to measure the significance of that computed correlation. For modeling and forecasting purposes it is often useful to plot the ACF and PACF against consecutive time lags. These plots help in determining the order of AR and MA terms. Some empirical considerations, indeed, relates to the following facts: if the ACF decreases slowly and the PACF decreases very quickly after p steps, the series is AR(p); if the ACF decreases very fast and the PACF decreases slowly after q step, the series is MA(q); if both the ACF and the PACF decrease slowly, the series is ARMA. By analyzing the graph shown in Figure 2, it is possible to see the ACF and PACF of a time series for 2 degrees of differentiation.

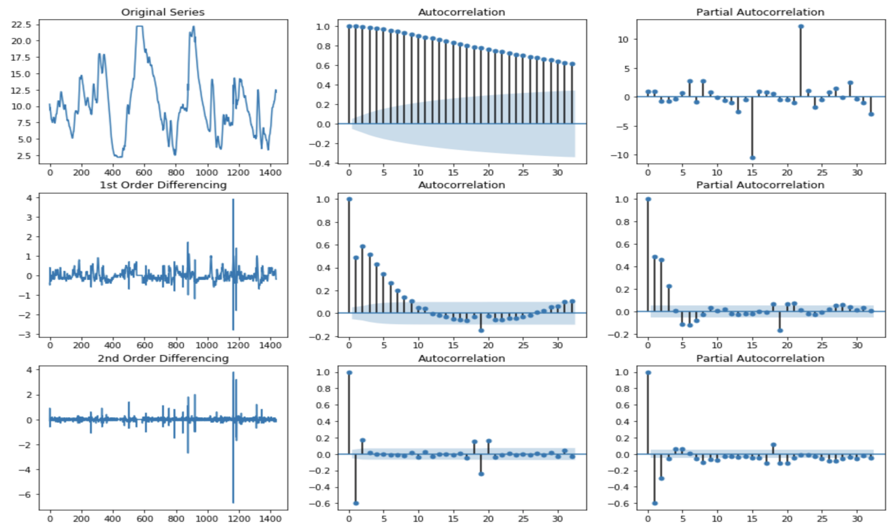

Fig. 2. Differentiated series, ACF and PACF

Auto adaptive ARIMA parameters optimization algorithm Given the previous considerations, in this research we define an adaptive logic for optimizing the parameters of the model p, q and d so that, for each new data point, the best model is recalculated in real-time on the basis of the historical data in an online learning fashion. In this way, the trend of the series is predicted for each new data point. Since prediction must be taken in real-time, it is important to quickly compute the selection of the parameters. The algorithm we designed works iteratively and is reported in pseudo-code in the following. At each iteration it takes the historical series (values) and calculates the differential of the series starting from 0 by incrementally increasing the value of the differential to be calculated. Once the differential has been calculated, the ADF statistical test is performed. This test verifies the null hypothesis that there is a root in the time series. The alternative hypothesis is the stationarity hypothesis. Through this test, it is therefore possible to determine with statistical significance whether the series is stationary or not. The test is verified by considering the value of the p-value and the value of the calculated ADF statistic. If the p-value < 0.05 and the calculated statistic is less than the first critical value returned by the test, then the series is considered stationary, otherwise the series is considered non-stationary. Once the test has been performed, if the null hypothesis holds, the degree of differentiation of the series is increased and the test is repeated, if the alternative hypothesis holds, instead the series is considered stationary and the algorithm proceeds in the determination of the parameters p and q. To determine such parameters, we use the calculation of the ACF and PACF functions. By following the empirical considerations previously described, to detect p and q we compute the cut-off of these series and take the first value of the function which is outside the confidence interval (considering an alpha of 0.05). We therefore consider the maximum value that p can take, p_max, as the cut-off value of the ACF function and the same for q, q_max, as the cut-off value of the PACF function, i.e. as the number of detections of the ACF and PACF functions that are outside the 95% confidence interval. The choice to exclude the values that are outside the confidence interval is determined by the fact that those values do not have an effective statistical significance in determining the trend of the series. For this reason, the hypothesis is that the p and q parameters should not be higher than these cut-off values, however it is not guaranteed that the cut-off values are the optimal model values. Indeed, once the p_max and q_max parameters have been determined, to guarantee the optimal choice of the model, a greedy search of the parameters is carried out, keeping d fixed and making p vary from 1 to the ACF cut-off value and q from 0 to the cut-off value of the PACF. For each model trained, the AIC is calculated. This parameter is an estimator of the prediction error and, consequently, of the relative quality of the statistical model calculated on the input data. Indeed, given a collection of models for a dataset, the AIC estimates the quality of each model relative to the quality of the others. In this way, the AIC provides an estimate to make the selection of the models. In this way it is possible to select the best possible model and with each new data the model can be retrained and used for prediction.

Algorithm 1: ARIMA Optimal Parameter Selection

```
Input: The time series values ts_val to train the ARIMA model
Output: The optimal parameter p, d, q of the ARIMA model for the given time series
d ← 0; check ← False;
while check=False do
    diff_val ← differentiate ts_val with order d;
    p_value, ADF_stat, critical_val ← compute ADF test over diff_val;
    if p_value<0.05 and ADF_stat<critical_val then
        acf_val ← compute ACF over diff_val;
        pacf_val ← compute PACF over diff_val;
        p_max ← compute cut off of acf_val;
        q_max ← compute cut off of pacf_val;
        check ← True;
    else
        d ← increment d by 1;
    end
end
best_p ← 0; best_q ← 0; aic_val ← ∞;
for all p̃ ∈ [1, p_max] do
    for all q̃ ∈ [1, q_max] do
        arima_model ← compute ARIMA(p̃, d, q̃);
        aic ← compute AIC for the arima_model;
        if aic < aic_val then
            best_p ← p̃;
            best_q ← q̃;
        end
    end
end
```

4.2 RNN model

An RNN is a class of artificial neural network in which the output values of a layer of a higher level are used as input to a layer of a lower level [30]. This interconnection between layers allows the use of one of the layers as state memory, and allows, by supplying a temporal sequence of values as input, to model a dynamic temporal behavior dependent on the information received at the previous time [31]. In the following, the functioning of the RNN is described in more detail. Consider an RNN with 3 layers: an input layer, a hidden layer and an output layer. The input of the network is a sequence of vectors collected at different moments in time. $\{..., x_{t-1}, x_t, x_{t+1}, ...\}, x_t = (x_1, x_2, ..., x_N)$. The input unit is connected to the hidden unit in the recurrent layer, where the connection is defined by a matrix of weights W. The hidden layer has M units $s_t = (s_1, s_2, ..., s_M)$ connected to each other over time through recurring connections. The hidden layer defines the system memory.

$$s_t = f(Ux_t + Ws_{t-1} + b_s) \tag{5}$$

In the above equation, f is the activation function of the hidden layer and b_s is the bias vector of the hidden unit. The output layer has P units $h_t = (h_1, h_2, ..., h_P)$ calculated as below, where V is the matrix of weights that connect the hidden layer with the output layer, g is the activation function of the output layer and b_h is the bias vector of the output unit.

$$h = g(Vs_t + b_h) \tag{6}$$

Thus, RNN represents a non-linear equation of state iterable over time. At any time, the hidden state provides a prediction of the output based on the input vectors [32]. In order to train an RNN it is not enough to use a gradient-based algorithm, such as Stochastic Gradient Descent, (SGD). This is because SGD is unable to take long-range time dependencies into account. This behavior is caused by the exponential decay of the gradient when it is propagated backwards and is known as a vanishing gradient problem. To remedy this situation, a variant of the SGD algorithm is used which avoids running into the vanishing gradient problem. This change concerns the propagation algorithm and is called backpropagation through time (BPTT). This propagation technique consists in unfolding the RNN so that it can be considered as a FFNN. However, if the network has to model very long time dependencies, this structure is not suitable, since the BPTT does not guarantee to avoid the vanishing gradient problem.

Long Short Term Memory model To overcome this limitation, architectures have been introduced that can store very long time dependencies. One of these is Long Short Term Memory (LSTM) [33]. The LSTM are RNNs that allow, thanks to their complex internal structure, to model longer temporal relationships without running into training problems due to the explosion of the gradient during updates [33]. This approach involves modifying the hidden structure by sigmoid or hyperbolic tangent to a memory cell, where the input and output are gate controlled. These gates control the flow of information to hidden neurons and preserve the features extracted at the previous instant. An LSTM cell at each instant t, comprises a status C_t and an output h_t. As input, the cell at time t includes x_t, C_{t-1} e h_{t-1}. Inside the LSTM cell, the computation is defined by gates (input, update, forget, output) that allow or not the transmission of information.

In order to define the best solution for the given problem, we decided to compare such LSTM-based models trained on Google Cloud Vertex AI leveraging a hyperparameter tuning tool based on Google Vizier known as Google HyperTune [28] with the ARIMA model with our auto adaptive parameter optimization algorithm. For the LSTM model Root Mean Squared Error (RMSE) is used as loss function and RMSprop as optimization algorithm.

5 Experimental results

As mentioned in Chapter 3, a public dataset (D1NAMO) is used to train and test the models. For all tested models, two prediction time horizons are compared, one at 30 minutes and one at 60 minutes. For both cases the training is carried out considering the measurements in the previous 8 hours. Thus, the time series is divided into sub-serieses of 8 hours and divided into train and test according to the 80-20 split. Since in the dataset the detections have a sampling period of 5 minutes, the prediction is made for 6 and 12 subsequent points and the training is done using 96 samples for each series. To measure the results we use RMSE defined as:

$$RMSE = \frac{1}{T}||\sum_{t=1}^{T}(\tilde{y}_t - y_t)^2|| \tag{7}$$

As far as the ARIMA model is concerned, it is trained with the last 96 samples of training data and over these the prediction is carried out (to be compared over the first 6 or 12 values of the test set). Then for each subsequent point with respect to the last 96 the ARIMA model is re-trained and the prediction is compared with the test data. This is done for 7 patient-related time series out of the 9 available, given the relative scarcity for 2 of these 9, and till all the test data are covered. Results are then averaged. As far as LSTM is concerned, we train a model for each patient over the training data, and each model is used to compute the prediction for subsequent values of the time series in the test set. Also in this case the results are averaged. Glucose values recorded in the reference dataset are measured in mmol/l, which has a conversion rate in mg/dl s of 18.02. Furthermore, given the need to make predictions in real time, training and prediction times are tracked in order to determine the most suitable solution also in terms of responsiveness. In Table 1 it is shown the RMSE at 30 minute and at 60 for both the trained models together with the average time necessary to train the models. As expected, the algorithms are more efficient in the case of shorter time horizons.

Table 1. Experimental results in terms of RMSE at 30 and 60 minutes and average training time

Algorithm	RMSE_30	RMSE_60	AVG Training Time
ARIMA with auto adaptive algorithm	1.11 mmol/l	2.23 mmol/l	0.5s
LSTM with Google Hypertune	2.75 mmol/l	3.13 mmol/l	106s

Compared to the ARIMA-based model, in general, the LSTM-based model is slightly less performing in terms of RMSE. This, as mentioned, is mainly due to the fact that the ARIMA-based model adapt at every moment in time to the historical series up to that current instant by extracting short or very short-term dependencies, while the LSTM-based model are static models, i.e. they do not change with each new detection, but try to extract longer-term dependencies. Moreover, for the evaluation of the model with a view to industrialization of the solution, as mentioned above, the training time is also relevant. Indeed, for each patient and for each new data point (which is estimated to be collected approximately every 3 minutes using the Glunovo CGM tool) the model is retrained considering the historical and the new detection in order to make a prediction in the next 30 or 60 minutes. For ARIMA, the average training time varies as the parameters p, d and q selected by the developed algorithm vary, but on average the measured value is 0.5 seconds, while for the LSTM the average training time si of 106 seconds. This means that the solution based on LSTM does not allow for real time training and prediction in an online learning fashion. Thus, to use that solution it would be necessary to train a model for each patient periodically, store it and use that model during prediction. This implies that a pipeline to manage models update is necessary. The solution based on ARIMA, on the other hand, allows not to store the model, but just to train it and to discard it after the prediction have been performed. Thus, the LSTM-based solution poses a major scalability problem, since as the number of patients increases, the number of models to re-train and manage will increase, resulting in increased storage and training costs. Therefore, even if the model is good in terms of performance, its industrialization is complex to maintain. The solution based on ARIMA, instead, allows to implement the model in an industrial environment, by training the model with each new collected data for each patient, guaranteeing the prediction in real time.

5.1 Infrastructural considerations

This study is carried out with the aim of being used by the pilot company within an application program developed on Google Cloud Platform that continuously monitors some vital parameters (such as glycemic index, the lipid profile and HbA1c, the blood pressure and heart rate) and that aims at predicting the trend of the glycemic index in order to raise alerts. The architecture of the machine learning module implemented is described in the following. Firstly, the data coming from the sensors are collected and sent to the time series forecasting module via Google Cloud PubSub [34]. This tool is an event-based message management tool. Each time a new message arrives, it is sent to a Google Cloud Function [35] which is registered at the specific reception event of messages from the sensors. Google Cloud Functions are computation tools used to create single-purpose and stand-alone functions that respond to cloud events (such as REST calls or, indeed, publication of a pub/sub message) without the need to manage a server or an environment runtime. Thanks to this tool, it is therefore possible to define a training and forecasting pipeline of the module described in this study and make this pipeline available as a fully managed service via API. In order to make predictions, however, it is necessary for each new message to extract all the previously recorded values for that patient. For this reason, the Cloud Function not only takes care of implementing the training and prediction logic of the model, but also of saving and extracting historical data. As a database for this kind of operation we use Google Cloud BigQuery [36]. BigQuery is a scalable and cost-effective DBMS optimized for data analysis. Specifically, the data saved on this DBMS relate to the individual measurements of the wearable devices, the morphological situation of the patients (weight, height, age), the patient data and the threshold values (upper and lower) within which every detection or prediction must be to avoid raising and alert. These threshold values are specific for each device and for each patient and depend on the patient's state of health: they are defined by a specialist doctor who uses the application to enter patient information and to monitor the summary dashboard of the progress of the various monitored indices of each patient.

6 Conclusion and future works

This research presents a comparison of a statistical and a deep learning based approach for the implementation of a glycemic index time series prediction, monitored by wearable devices. We compare two state of the art approaches: one based on the implementation of statistical ARIMA models by designing an algorithm to automatically identify the value of the optimal parameters of the model based on the use of the ACF and PACF, so that at each iteration the most accurate model possible is created in an online fashion; one based on LSTM with the aim of creating a single prediction model for each monitored patient by taking longer histories into consideration, providing for periodic offline retraining. We used a public dataset to train and test these models which contained real blood glucose monitoring data for different patients. Tests were conducted considering the RMSE with a 30-minute and 60-minute horizon and the time necessary to train and optimize the algorithm. The training was done considering the series in the previous 8 hours. The model based on ARIMA, with our custom parameters optimization algorithm, is the most performing in the 30 and 60 minutes with a RMSE_30 of 1.11 mmol/l and a RMSE_60 of 2.23 mmol/l. Additionally, this model has an average training time of 0.5 seconds which makes it a good candidate for industrialization. On the other hand, if the LSTM-based model has good performances in terms of RMSE (albeit lower than that of the ARIMA-based model), the serving logic of these models is complex: it would be necessary to train a model for each patient with the consequent explosion of the resources necessary for saving, serving and periodic re-training of these models. Thus, we show that the ARIMA-based model can be used in a production environment in an online learning fashion guaranteeing good performances. However, some enhancement can be applied. Indeed, recent studies show how it is possible to increase the performance of glucose level prediction algorithms by considering further data as input, such as information about meals consumed by patients, the doses of insulin taken and the physical exercise performed. All this information can thus be used to improve time series prediction performance. Moreover, in future study, we intend to test the proposed auto adaptive algorithm also in other time series forecasting contexts, like sales forecasting, to test its generalizability.

References

1. Ogurtsova, K., da Rocha Fernandes, J. D., Huang, Y., Linnenkamp, U., Guariguata, L., Cho, N.H., Cavan, D., Shaw, J., Makaroff, L: IDF Diabetes Atlas: Global estimates for the preva-

lence of diabetes for 2015 and 2040. Diabetes Research and Clinical Practice **128**, 40–50 (2017)

2. Béranger, J.: The Shift towards a Connected, Assessed and Personalized Medicine Centered Upon Medical Datasphere Processing. Elsevier (2016)
3. Swan, M.: The quantified self: Fundamental disruption in big data science and biological discovery. Big data **1**(2), 85–99 (2013)
4. Adhikari, R., Ramesh, K. A.: An introductory study on time series modeling and forecasting. Lambert Academic Publishing, Germany (2013)
5. Bittanti, S.: Identificazione dei modelli e sistemi adattivi. 3rd edn. Pitagora, Italy (2005)
6. Lunze, K., Singh, T., Marian, W., Brendel, M. D., Leonhardt, S.: Blood glucose control algorithms for type 1 diabetic patients: A methodological review, Biomedical signal processing and control **8**(2), 107–119 (20013)
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 **33**(6) (2017)
8. Woldaregay, A., Arsand, E., Walderhaug, S., Albers, D., Mamykina, L., Botsis, T., Hartvigsen, G.: Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes. Artificial intelligence in medicine **98**, 109–134 (2019)
9. Bock, A., François, G., Gillet, D.: A therapy parameter-based model for predicting blood glucose concentrations in patients with type 1 diabetes. Computer methods and programs in biomedicine **118**(2), 107–123 (2015)
10. Pappada, S., Cameron, B., Rosman, P., Bourey, R., Papadimos, T., Olorunto, W., Borst, M.: Neural network-based real-time prediction of glucose in patients with insulin-dependent diabetes. Diabetes technology and therapeutics **13**(2), 135–141 (2011)
11. Eren-Oruklu, M., Cinar, A., Rollins, D., Quinn, L.: Adaptive system identification for estimating future glucose concentrations and hypoglycemia alarms. Automatica **48**(8), 1892–1897 (2012)
12. Li, K., Daniels, J., Liu, C., Herrero, P., Georgiou, P.: Convolutional Recurrent Neural Networks for Glucose Prediction. IEEE Journal of Biomedical and Health Informatics, **24**(2), 603–613, (2020)
13. Bunescu, R., Struble, N., Marling, C., Shubrook, J., Schwartz, F.: Blood Glucose Level Prediction Using Physiological Models and Support Vector Regression. In: 2013 12th International Conference on Machine Learning and Applications, pp. 135–140, IEEE, Miami, FL, USA (2013)
14. van Heusden, K., Dassau, E., Zisser, H.C., Seborg, D.E., Doyle, F.J.: Control-relevant models for glucose control using a priori patient characteristics. IEEE Transactions on Biomedical Engineering **59**(7), 1839–1849 (2012)
15. Toffanin, C., Del Favero, S., Aiello, E., Messori, M., Cobelli, C., Magni, L.: Glucose-insulin model identified in free-living conditions for hypoglycaemia prevention. Journal of Process Control **64**, 27–36 (2018)
16. Toffanin, C., Aiello, E., Del Favero, S., Cobelli, C., Magni, L.: Multiple models for artificial pancreas predictions identified from free-living condition data: A proof of concept study. Journal of Process Control **77**, 29–37 (2019)
17. Fernandez de Canete, J., Gonzalez-Perez, S., Ramos-Diaz, J.C.: Artificial neural networks for closed loop control of in silico and ad hoc type 1 diabetes. Computer methods and programs in biomedicine **106**(1) , 55–66 (2012)
18. Shah, M., Franklin, B., Adams-Huet, B., Mitchell, J., Bouza, B., Dart, L., Phillips, M.: Effect of meal composition on postprandial glucagon-like peptide-1, insulin, glucagon, C-peptide, and glucose responses in overweight/obese subjects. European journal of nutrition **56**(3), 1053–1062 (2017)

19. Gani, A.,Gribok, A.V., Lu, Y., Ward, W.K., Vigersky R.A., Reifman, J.: Universal Glucose Models for Predicting Subcutaneous Glucose Concentration in Humans. IEEE Transactions on Information Technology in Biomedicine **14**(1), 157–165 (2010)
20. Montaser, E., Díez, J-L. Bondia, J.: Stochastic seasonal models for glucose prediction in the artificial pancreas. Journal of diabetes science and technology **11**(6), 1124–1131 (2017)
21. Zarkogianni, K., Mitzis, K., Litsa, E., Arredondo, M.T., Fico, G., Fioravanti, A., Nikita, K.S.: Comparative assessment of glucose prediction models for patients with type 1 diabetes mellitus applying sensors for glucose and physical activity monitoring. Medical and biological engineering and computing **53**(12), 1333–1343 (2015)
22. Hewamalage, H., Bergmeir, C., Bandara, K.: Recurrent neural networks for time series forecasting: Current status and future directions. International Journal of Forecasting **7**(1), 388-427 (2021)
23. Shanthi, S., Kumar: A novel approach for the prediction of glucose concentration in type 1 diabetes ahead in time through ARIMA and differential evolution. Advanced engineering Informatics, **38**, 4182-–4186 (2011)
24. Ying, Z., Rui, K., Shihong., X.: Research on glucose concentration predicting based on ARMA model. In: 2014 prognostics and system health management conference (PHM-2014 Hunan), pp. 332–335, IEEE, Zhangjiajie, China (2014)
25. Yang, J., Li, L., Shi, Y., Xie, X.: An ARIMA model with adaptive orders for predicting blood glucose concentrations and hypoglycemia. IEEE journal of biomedical and health informatics **23**(3), 1251–1260 (2019)
26. Martinsson, J., Schliep, A., Eliasson, B., Mogren, O.: Blood glucose prediction with variance estimation using recurrent neural networks. Journal of Healthcare Informatics Research **4**(1), 1–18 (2020)
27. Allam, F., Nossai, Z., Gomma, H., Ibrahim, I., Abdelsalam Besibes, M.: A Recurrent Neural Network Approach for Predicting Glucose Concentration in Type-1 Diabetic Patients. IFIP Advances in Information and Communication Technology **363**, 254–259 (2011)
28. Golovin, D.,Solnik, D., Moitra, S., Kochanski, G., Karro, J., Sculley, D.: Google Vizier: A Service for Black-Box Optimization. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17), pp. 1487–1495. Association for Computing Machinery, New York,USA (2017)
29. Dubosson, F., Ranvier, J-E., Bromuri, S., Calbimonte, J-P., Ruiz, J., Schumacher, M.: The open D1NAMO dataset: A multi-modal dataset for research on non-invasive type 1 diabetes management. Informatics in Medicine Unlocked **13**, 92–100 (2018)
30. alehinejad, H., Sankar, S., Barfett, J., Colak, E., Valaee, S.: Recent advances in recurrent neural networks. ArXiv preprint, arXiv:1801.01078 (2017)(2017).
31. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**, 436–444 (2015)
32. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press, (2016)
33. Hochreiter, S., Schmidhuber, J.: Long Short-term Memory. Neural computation **9**, 1735–1780 (1997)
34. Google Cloud PubSub Documentation, https://cloud.google.com/pubsub/docs. Last accessed 31 Aug 2021
35. Google Cloud Function Documentation, https://cloud.google.com/functions/docs. Last accessed 31 Aug 2021
36. Google Cloud BigQuery Documentation, https://cloud.google.com/bigquery/docs. Last accessed 31 Aug 2021