J Med Syst (2017) 41: 142
DOI 10.1007/s10916-017-0788-2

PATIENT FACING SYSTEMS

Data Based Prediction of Blood Glucose Concentrations Using Evolutionary Methods

J. Ignacio Hidalgo[1] · J. Manuel Colmenar[2] · Gabriel Kronberger[3] · Stephan M. Winkler[3] · Oscar Garnica[1] · Juan Lanchares[1]

Received: 8 May 2017 / Accepted: 20 July 2017 / Published online: 8 August 2017

Abstract Predicting glucose values on the basis of insulin and food intakes is a difficult task that people with diabetes need to do daily. This is necessary as it is important to maintain glucose levels at appropriate values to avoid not only short-term, but also long-term complications of the illness. Artificial intelligence in general and machine learning techniques in particular have already lead to promising results in modeling and predicting glucose concentrations. In this work, several machine learning techniques are used for the modeling and prediction of glucose concentrations using as inputs the values measured by a continuous monitoring glucose system as well as also previous and estimated future carbohydrate intakes and insulin injections. In particular, we use the following four techniques: genetic programming, random forests, k-nearest neighbors, and grammatical evolution. We propose two new enhanced modeling algorithms for glucose prediction, namely (i) a variant of grammatical evolution which uses an optimized grammar, and (ii) a variant of tree-based genetic programming which uses a three-compartment model for carbohydrate and insulin dynamics. The predictors were trained and tested using data of ten patients from a public hospital in Spain. We analyze our experimental results using the Clarke error grid metric and see that 90% of the forecasts are correct (i.e., Clarke error categories A and B), but still even the best methods produce 5 to 10% of serious errors (category D) and approximately 0.5% of very serious errors (category E). We also propose an enhanced genetic programming algorithm that incorporates a three-compartment model into symbolic regression models to create smoothed time series of the original carbohydrate and insulin time series.

Keywords Diabetes · Glucose prediction · Continuos glucose monitoring · Evolutionary computation

This article is part of the Topical Collection on *Patient Facing Systems*

✉ J. Ignacio Hidalgo
hidalgo@ucm.es

J. Manuel Colmenar
josemanuel.colmenar@urjc.es

Gabriel Kronberger
Gabriel.Kronberger@fh-hagenberg.at

Stephan M. Winkler
Stephan.Winkler@fh-hagenberg.at

Oscar Garnica
ogarnica@ucm.es

Juan Lanchares
julandan@ucm.es

[1] Adaptive and Bioinspired System Group, School of Informatics, Universidad Complutense de Madrid, C/ Profesor José García Santesmases 9, 28040, Madrid, Spain

[2] Universidad Rey Juan Carlos, C/ Tulipán s/n, 28933, Móstoles, Spain

[3] Heuristic and Evolutionary Algorithms Laboratory, University of Applied Sciences Upper Austria, Softwarepark 11, 4232 Hagenberg, Austria

Introduction

Diabetes is a chronic disease that affects an increasing number of people in the developed countries. According to the International Diabetes Federation [22], more than 387 million people suffered from diabetes worldwide in 2014. The illness is a complex disease produced by a defect either in the production or the action of insulin, a hormone produced

by the pancreatic system; insulin is necessary for regulating the absorption of glucose by several important types of cells of the human body. Most of the diabetic patients can be classified into two main types: Type 1 Diabetes Mellitus (T1DM) and Type 2 Diabetes Mellitus (T2DM) patients. People with T1DM suffer from an autoimmune disease which attacks the cells of the pancreas that are in charge of the production of insulin. In other words, pancreas of T1DM patients do not produce enough insulin to process the sugar in the blood. Depending on the level of evolution of the disease, sometimes they have some cells able to produce insulin. However, with the advance of the disease, the production is eventually terminated. On the other hand, patients with T2DM produce insulin, but their bodies develop a resistance to it so that it has little or no effect. In both cases the result is an increase of the levels of glucose in the bloodstream. Maintaining high values of glycemia during long periods of time may cause other chronic diseases like blindness and brain or kidney injuries. The already mentioned long-term complications can be dangerous, and there are also risks related to short term complications due to low glucose levels: hypoglycemia happens when the glucose drops below a certain threshold, which might cause unconsciousness or even cause the death of the patient.

For these reasons, diabetics must control their blood glucose levels throughout their lives, trying to keep them at reasonable levels, similar to those of a subject who does not suffer from the disease. This is quite complicated, especially for T1DM diabetics. The concept commonly used to define the quality of control of glycemia is the "time in range" value, which is defined as the time over which a diabetic patient's glucose level is in the recommended range. Usually, this range is [70–180] mg/dl [43]. When an insulin-dependent diabetic is going to have a meal, he or she needs to estimate the units to be injected so that, after the meal, the glucose levels are kept within a healthy range. This estimation must be made based on many factors, but mainly, the patient knows his / her glucose value at that moment and estimates the amount of food eaten, usually measured in carbohydrate rations. In other words, blood glucose control in T1DM patients requires the prognosis of future glucose values on the basis of the amount of food intakes, injection of insulin, and/or glucagon. As the reader can deduce, from a scientific point of view, this process involves many estimations and is not a clearly defined process in the variables involved. Fortunately, recent advances in both, devices and algorithms, allow the automation of some parts of this control process and facilitate the task for diabetics.

Depending on the reached degree of automation, we currently identify three different kinds of blood glucose control strategies (see Table 1): traditional therapies with manual calculation and administration of the insulin protocol (by hand) [24], insulin pump therapies (semi-automated) [46], and solutions based on the artificial pancreas [6]. In the case of traditional therapies, decisions regarding the administration of insulin are made entirely by the patients under the guidance of medical staff and their own experience. During the day they need to measure glucose levels and take decisions on the administration of multiple manual insulin injections and/or on eating some source of carbohydrates to correct low values of blood glucose. Continuous glucose monitoring systems (CGMS) can be valuable here by measuring the value of glucose and reducing the number of punctures with standard glucometers. Insulin pump therapies include the use of continuous subcutaneous insulin infusion systems (CSIIS), also known as insulin pumps, and sometimes of CGMS [46]. CGMS help to automate some decisions such as stopping the pump in dangerous situations. However, in most of the therapies, patients still need to be alert and to detect anomalous glucose situations, stopping the infusion of insulin or correcting a trend through the infusion of glucagon or additional meals. The decision of the amount of insulin given before the meals is done in the same way as in traditional therapies. Using an artificial pancreas (AP) could lead the way to an ideal solution [12]. The main components of an AP are, as shown in Fig. 1: a CGMS, glucose control algorithms, models to calculate the amount of insulin needed, and an insulin pump. APs can also include bi-hormonal infusion pumps. Even though results that show the advance on the research of APs and their successful use also in clinical studies have been published recently [27], the independence of patients with AP cannot be guaranteed at today's stage of development. Even though some automation is implemented in these systems, the patient (or the automated system) needs to predict future glucose values to take treatment decisions.

In this paper, we propose and test several artificial intelligence techniques to model and predict glucose taking into account food intakes, insulin injections, and past glucose values. One of the main reasons why this is a hard problem lies in the different delays and effects that the food intakes and the hormones have among different patients, and also in the fact that there can be variable effects for the same patient in different periods of time. Moreover, the behavior of the patients' body also changes with environmental conditions, age, stress and other eventual situations. Our proposal is to apply several machine learning techniques in order to obtain customized models for the prediction of the glucose level for each patient. In this way, we will follow an approach where the models will be extracted from the data. We will first run a training phase over a set of data from patients that will produce a number of models that will be evaluated in a test phase with data from the same patients. We here compare four different techniques: genetic programming (GP), random forest regression (RF), k-nearest neighbors (KNN), and grammatical evolution (GE). We contribute with a

Table 1 Classification of glucose control strategies in the treatment of T1DM. CGMS calibration is only required when the control strategy uses a continuous glucose monitoring system

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

methodology to study several approximations developed by GP and GE. RF regression, GP, and KNN were implemented in HeuristicLab [30], while GE was implemented using the JECO library [2]. The predictors were trained and tested using data of ten patients from a public hospital in Spain. For the analysis of our experimental results we use the Clarke error grid (CEG) metric [9].

The rest of the paper is organized as follows: Section "State of the art in modeling glycemia" reviews the state of the art in glucose modeling and prediction. Section "Heuristics for prediction" explains the different approaches and details the main contributions of this works. Section "Experimental results" shows the experimental results performed on data of ten diabetic patients from a public hospital in Spain. We conclude the paper in Section "Conclusion" and propose future work.

State of the art in modeling glycemia

Currently, several attempts in building an AP have been made by different companies, and promising results have been reported in clinical studies with patients [21]. However, still there is not a completely safe process around AP alternatives. There is a wide variety of modeling options to predict future blood glucose values. According to the review presented in [38], we can define three categories for prediction models: physiological models, data-driven models, and hybrid models (cf. [32]). In this paper we use different approaches from the last two categories. Physiological modeling (PM) was the most common approach some years ago, however, there are issues when using this method since an *a priori* structure of the models must be assumed. In addition, as [38] pointed out, PM requires previous knowledge to set the physiological constants. In contrast, data-driven models are often hard to interpret since they do not fix any physiologically inspired mathematical formulation, therefore the model does not represent any known physiological behaviour. Combining both, scientists are also looking for hybrid models. These models take the simpler parts of the PM that are completely tested and include data to determine the parameters of the models. The review in [38] also highlights that many proposals do not use data from patients; instead, frequently researchers work with *in-silico* patients generated using different simulators. In addition, the authors suggest that the studies should compare different prediction techniques with the same data from real patients.

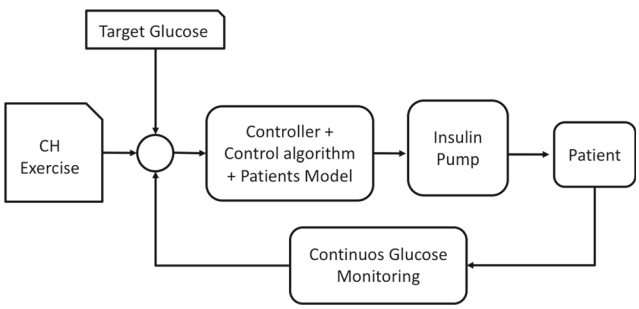

Fig. 1 A schematic view of an artificial pancreas system

Zhao et al. have used multiple algorithms for data-driven modelling for predictions for *in-silico* patients as well as for real patients [48]. They have shown that, using a latent-variable based statistical method, approximately 72% of the samples can be predicted correctly (i.e. are in Clarke Error category A) for horizons of up to 60 minutes. For the *in-silico* patients significantly better results could be achieved with up to 90% of correctly predicted samples. Another study shown in [23] reported an RMSE of 0.7 mmol/l for a prediction horizon of 60 minutes and 1.6 mmol/l for a prediction horizon of 90 minutes; here, data of nine simulated subjects using autoregressive models have been used.

Another review of different methods to predict blood glucose levels is presented in [11], where the authors focus on physiological models and analyze methods based on continuous glucose monitors that generate time series data. This work does not mention more recent techniques related to meta-heuristics and is more focused on systems that could be directly implemented as wearable or portable devices for patients. For decades, many works have used GP for time series prediction problems (see, e.g., [41] or [18]), but few works have applied these techniques to glucose prediction so far. Nevertheless, several approaches for glucose prediction have used GE. Contreras et al. obtained good predictions for 60 minutes windows using real patients data [17]. This method is based on a previous paper by Hidalgo et al. [25], where *in-silico* patients data were studied. Recently we

presented a work where both GP and GE techniques were applied to the glucose prediction problem [14].

Our new proposal presents several novelties regarding both GP and GE techniques. In the case of GP, we have incorporated a new compartment model prediction strategy in combination with a specific data preprocessing stage. In GE we also use this new data treatment method and developed more effective grammars that are specifically suited for this problem. Moreover, we have also used RF and KNN and compare the performance of all these techniques using the same data set from real patients.

Heuristics for prediction

In this section, we discuss the methods that we used for the prediction of blood glucose concentration values. We consider a forecasting window of two hours. The data are available as a time-series with observations in a regular interval of five minutes. There are several kinds of models designed for time-series prediction that could be used here, such as for example the ARIMA model [28]. However, we have transformed this time-series prediction problem into multiple independent regression problems where the goal is to find four independent models for the prediction of blood glucose concentration in 30, 60, 90, and 120 minutes. By transforming the time-series prediction problem in this way we can apply any regression technique including symbolic regression using GP and GE. A drawback of this transformation is that the models do not allow us to make predictions about any other point in the forecasting window. However, this is not a problem from a practical point of view, since for diabetic patients, it is sufficient to predict values for four distinct points in the forecasting window. The advantage is that we can use data from the continuos glucose monitoring systems directly to make insulin recommendations based on the four time horizons, by using the equations given by the obtained models. GP and GE are indicated for solving symbolic regressions problems.

Our goal is to develop the following models:

$$\begin{aligned}
\hat{G}_{t+30} &= f_{t+30}(G_{t+i}, i \in (-240 \ldots 0)), \\
&\quad (I_{t+i}, C_{t+i}), i \in (-240 \ldots +30)) \\
\hat{G}_{t+60} &= f_{t+60}((G_{t+i}, i \in (-240 \ldots 0)), \\
&\quad (I_{t+i}, C_{t+i}, i \in (-240 \ldots +60)) \\
\hat{G}_{t+90} &= f_{t+90}((G_{t+i}, i \in (-240 \ldots 0)), \\
&\quad (I_{t+i}, C_{t+i}, i \in (-240 \ldots +90)) \\
\hat{G}_{t+120} &= f_{t+120}((G_{t+i}, i \in (-240 \ldots 0)), \\
&\quad (I_{t+i}, C_{t+i}, i \in (-240 \ldots +120))
\end{aligned}$$

where the time offsets are given in minutes i.e. we are forecasting blood glucose $\hat{G}$ up to 120 minutes ahead. G is the time series of measured blood glucose concentration values, I is the time series of insulin inputs (pump and bolus), and C is the amount of carbohydrate inputs as estimated by the patients. At each time point t data from up to four hours before are available for prediction. To generate the four prediction models f_{t+30}, f_{t+60}, f_{t+90}, f_{t+120} we used KNN regression, RF regression, and enhanced variants of GE as well as tree-based GP for evolving symbolic regression models and compared our results with two simple base-line predictions. For GE as well as tree-based GP we made application-specific enhancements which are described in detail in Sections "Symbolic regression by tree-based genetic programming" and "Modeling glycemia by grammatical evolution with data preprocessing and a glucose specific grammar".

Data preprocessing

In a preprocessing stage, we aggregate the data over time periods using the following function:

$$avg(X, t, \text{range}) = \frac{\sum_{i \in \text{range}}(X_{t+i})}{|\text{range}|} \tag{1}$$

where X is a time series, and *range* is a range of time offsets. The objective is to reduce the number of features involved in the model search. For instance, if we have four values of X in the period $[t_1, t_2]$, $|\text{range}| = 4$ and

$$avg(X, t, [t_1, t_2]) = \frac{\sum_{i \in 1\_to\_4}(X_{t+i})}{4} \tag{2}$$

and instead of using 4 features we use just one represented by $avg(X, t, [t_1, t_2])$. For time t we define the following set of features $\mathcal{F}(t)$ that describe the history of the time series (glucose, insulin, and carbohydrates) until t as well as the future intakes of insulin and carbohydrates:

$$\begin{aligned}
\mathcal{F}(t) \;=\; & \mathcal{F}_{\text{history}}(G, t) \cup \mathcal{F}_{\text{history}}(I, t) \cup \mathcal{F}_{\text{history}}(C, t) \cup \\
& \mathcal{F}_{\text{future}}(C, t) \cup \mathcal{F}_{\text{future}}(I, t) \cup \\
& \{C(t), I(t), G(t)\}
\end{aligned} \tag{3}$$

$$\begin{aligned}
\mathcal{F}_{\text{history}}(X, t) = \{ & avg(X, t, [-15, 0[), avg(X, t, [-30, -15[), \\
& avg(X, t, [-45, -30[), avg(X, t, [-60, -45[), \\
& avg(X, t, [-90, -60[), avg(X, t, [-120, -90[)\}
\end{aligned} \tag{4}$$

$$\begin{aligned}
\mathcal{F}_{\text{future}}(X, t) = \{ & avg(X, t,]0, 15]), avg(X, t,]15, 30]), \\
& avg(X, t,]30, 45]), avg(X, t,]45, 60]), \\
& avg(X, t,]60, 75]), avg(X, t,]75, 90]), \\
& avg(X, t,]90, 105]), avg(X, t,]105, 120])\}
\end{aligned} \tag{5}$$

We used this set of features for predictions with KNN, RF and standard tree-based GP. For GE we use in some cases some of this features, while for the enhanced GP approach we only used the three time series without averaging. We selected those aggregations because time steps of 15 minutes are interesting, since that is the time necessary to detect changes on glucose values in the blood with respect to the value measure by the sensor.

k-nearest neighbours regression

KNN [5] is a so-called lazy learning approach as it does not train a model that is later used for prediction but instead calculates predictions based solely on the data. For a given set of input values the k most similar situations in the training data are determined and the prediction value is calculated as an average value of the actually observed values found in the training data set. For the determination of similarity most frequently a metric space with Euclidean distance is assumed. The most important parameter for KNN is the number of observations k used for the calculation of the prediction value where small k values induce a high variance of predictions while high k values induce a large bias of predictions. It is important to note that the scaling of features for KNN regression has a large impact on the similarity calculation and therefore on the quality of predictions. Ideally, the scaling of variables should be derived from a deep understanding of the application but often this is not possible *a priori*. Therefore, in our experiments we optimized variable scaling coefficients using a simulated annealing procedure. The parameter k was determined using grid search and cross-validation.

As a model-free approach for forecasting glucose values at time stamps $t_{\texttt{future}} > t$ we look for similar situations, i.e. data snapshots that can be considered similar to the data at time t. In detail, we look for similar situations that were recorded previously, for which we recorded similar current and past glucose levels, past and future carbohydrate intakes, and past and future insulin injections.

First, before similar situations can be searched we normalize all features X to mean 0.0 and variance 1.0:

$$X_{norm} = (X - avg(X))/std(X) \tag{6}$$

where avg() represents averaged values and std the standard deviation of them. The distance of two situations at times t_1 and t_2 (i.e. data samples $s(t_1)$ and $s(t_2)$) can be calculated as the weighted squared difference of the samples' feature values, the difference according to each feature is weighted with a factor w_i:

$$\begin{aligned} dist(s(t_1), s(t_2), w) &= dist(\mathcal{F}(t_1), \mathcal{F}(t_2), w) \\ &= \frac{\sum w_i * (\mathcal{F}_{norm}(t_1)[i] - \mathcal{F}_{norm}(t_2)[i])^2}{\sum_i w_i} \end{aligned} \tag{7}$$

where w is a vector of weight factors w_i. Given a set of training data time stamps TR we look for the k nearest neighbor time stamps ($nn(t, w, k, TR)$) for sample $s(t)$ as those that have a smaller distance to the target sample than the other samples:

$$\begin{aligned} &nn(t, w, k, TR) : |nn(t, w, k, TR)| = k \wedge \\ &\forall_{i,j \in TR} : (i \in nn(t, w, k, TR) \wedge j \notin nn(t, w, k, TR)) \\ &\quad \Rightarrow dist(s(t), s(i), w) < dist(s(t), s(j), w) \end{aligned} \tag{8}$$

The future glucose progress from time t on is denoted as $G_{\text{future}}(t)$. The estimated future glucose progress $\hat{G}_{\text{future}}$ is the average of the progresses of the similar situations, where each progress is shifted by an offset o_i that is calculated as the difference of $G(t)$ and the first glucose value of the progress:

$$G_{\text{future}}(t) = [G(t+i), i \in \{0, \ldots 120\}] \tag{9}$$

$$\begin{aligned} \hat{G}_{\text{future}}(t, w, k, TR) = \frac{1}{k} * \sum_{j=1}^{k} \big(& G_{\text{future}}(nn(t, w, k, TR)_j) \\ & + (G(t) - G(nn(t, w, k, TR)_j)) \big) \end{aligned} \tag{10}$$

Figure 2 shows an example: The glucose values are known until time t, and the three most similar time series for $s(t)$ are identified. The average of these nearest neighbors' progresses is calculated as the prediction $\hat{G}_{\text{future}}(t)$.

The quality of a prognosis for time t can thus be defined as

$$\begin{aligned} &quality(\hat{G}_{\text{future}}(t, w, k, TR)) \\ &\quad = \frac{1}{n} \sum_{i=1}^{n} ((\hat{G}_{\text{future}}(t, w, k, TR)_i - G_{\text{future}}(t)_i)^2) \end{aligned} \tag{11}$$

Using these definitions we can define the calculation of the quality of a combination of the number of nearest neighbors k and the weightings set w as

$$\begin{aligned} &quality(w, k, TR, \text{VAL}) \\ &\quad = \frac{1}{|\text{VAL}|} \sum_{t \in \text{VAL}} quality(\hat{G}_{\text{future}}(t, w, k, TR)) \end{aligned} \tag{12}$$

where VAL is a set of validation data time stamps. For the experiment documented in this paper we optimized k and w using a $(\mu + \lambda)$ evolution strategy [40, 42]: initially, μ solution candidates $[k, w]$ are created with $k \in \{1 \ldots 10\}$ and w_i drawn from $\mathcal{N}(1, 0.5)$. In each generation we create λ solution candidates by drawing randomly from the generation and mutating the chosen parent solution; each weight w_i of the mutant is calculated as the parent's w_i plus a value drawn from $\mathcal{N}(0, \sigma)$, and k is either reduced by one, increased by one, or remains unchanged. The mutation strength parameter σ is in each generation adapted in dependence of the mutations' success: σ is increased 10% if more

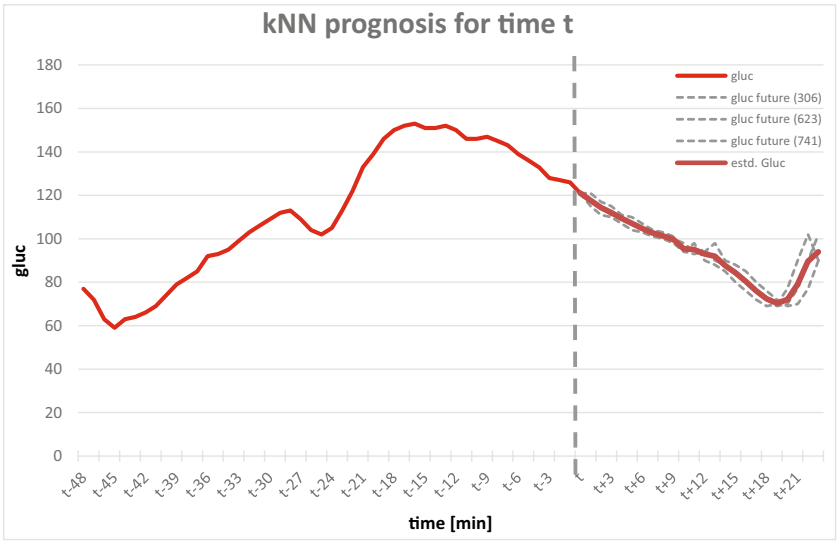

Fig. 2 Exemplary calculation of $\hat{G}_{\text{future}}(t)$ as the average of the three nearest neighbors' progresses

than 20% of the mutations lead to better solution quality, and σ is decreased 10% if the ratio of successful mutations is below 20%. In each generation, the μ best individuals are drawn from the λ new solution candidates and the previous generation's candidates.

We use the identifier KNN to refer to this configuration of k-nearest-neighbour regression in the following sections.

Random forest regression

In RF regression a large number of individual regression trees are combined to an ensemble, whereby the individual trees are trained on random subsets of the full training set [8]. RF regression uses the principle of bagging to combine multiple models into an ensemble whereby the output of the ensemble is calculated as the average of the outputs of the individual models. The individual models are complex and overfitted on purpose, as the averaging operation has the effect of stabilizing the output. Even though the high complexity of individual models is necessary to produce good predictions it has drawbacks such as high computational effort for model evaluation and high memory requirements.

Random forests have become very popular for real-world applications [7];[1] as only a few parameters have to be tuned and RFs often produce very accurate predictions, they can be used effectively even without in-depth expert knowledge. We have used the random forest implementation of *alglib*[2] and tuned the parameters r the fraction of rows and m the fraction of columns which are chosen randomly for training each individual tree. We have set the number of trees to 100 and have used cross-validation with grid-search to determine the parameter values for r and m. For both parameters we have allowed values from 10 to 70% in steps of 10%. Optimal values have been determined based on the cross-validated root of mean of squared errors. We use the identifier RF to refer to this configuration of random forest in the following sections.

Symbolic regression by tree-based genetic programming

GP is a method for automated synthesis of computer programs based on the concept of evolutionary computation [31]. It uses principles of natural evolution and simulates natural selection, breeding and random mutation to evolve a population of computer programs starting from a randomly initialized population. The objective is to find a computer program that solves a given programming problem usually specified using a set of test cases by simulating many generations. One task where genetic programming has proven to be particular effective is symbolic regression. In symbolic regression the objective is to identify a prediction model for a real-valued target variable whereby the model is represented as a mathematical formula. The structure of the model as well as its parameters must both be determined by the solution method to fit the given dataset.

We used the implementation for symbolic regression based on genetic programming which is available in the open-source software framework HeuristicLab[3] [30]. For the terminal set we used all pre-calculated features as described above as well as constants; each feature is accompanied with a real-valued weight which is initialized randomly using a Normal distribution $N(\mu = 1, \sigma = 1)$ and mutated randomly by adding a value sampled from

[1] Also see Howard and Bowles: "The Two Most Important Algorithms in Predictive Modeling Today", STRATA Conference, 2012, O'Reilly.

[2] http://www.alglib.net/

[3] http://dev.heuristiclab.com/

$N(\mu = 0, \sigma = 0.05)$. Real-valued constants are initialized randomly using a uniform distribution $U(-20.0, 20.0)$, and the mutation of constants adds a value sampled from $N(\mu = 0, \sigma = 1.0)$. The function set used here is

$$\{+, -, \times, /, log(x), exp(x)\}$$

where the protected variants of division and the logarithmic function are used [31]. The models in the initial population (population size: 100) have been generated using the probabilistic tree creator (PTC2) [33] using a maximum depth limit of eight levels and a maximum size limit of 100 nodes. The same depth and size constraints are also enforced in crossover and mutation operations. For crossover, we used a subtree crossover operator while for mutation we used a set of different operators which either replace a whole subtree with a randomly initialized tree, mutate all nodes of the tree or mutated only a single random node of the tree. A randomly selected mutation operator is executed with a probability of 25% (mutation rate) after each crossover operation.

We used strict offspring selection [3] in combination with fitness proportional parents selection. When using strict offspring selection, offspring are only accepted if they are better than both parents, other offspring are discarded. This has the effect that multiple repeated tries of parents selection, crossover, optional mutation and fitness evaluation are necessary until one offspring is added the population of the next generation (cf. [4]). The selection pressure threshold for stopping the algorithm was set to 100 and the maximum number of generations was set to 25.

The fitness of solution candidates is calculated as the coefficient of determination (Pearson's R^2) between the target blood glucose concentration values and the model outputs (see [29]). The finally selected prediction models are scaled linearly to minimize the sum of squared errors between target values and model outputs. We have not tuned GP parameters specifically for this modeling task, but we used robust standard settings. Cross-validation was executed in the same way as for all other supervised learning methods to calculate the cross-validated error values which are reported at the end of Section "Experimental results". We use the identifier GP and GP_gen to refer to this configuration of tree-based genetic programming for symbolic regression in the following sections.

Symbolic regression by tree-based genetic programming and compartment models

In addition to the rather general variant of tree-based GP for symbolic regression which we have described in the previous section, we have also implemented an enhanced variant of tree-based GP especially tailored for the task for blood glucose prediction. The underlying idea is to extend the function set by specific functions so that it is possible to work directly with the original time series of measured glucose and insulin values and reported carbohydrate consumption instead of the manually crafted features described above. The idea is motivated by existing hand-crafted models of insulin and blood glucose dynamics in the human body [47]. It is important to note that bolus of insulin represent an almost instantaneous impulse to the system, but the uptake of insulin and the degradation of insulin are much slower processes, so that the dynamics of blood insulin are gradual and smooth. The uptake of carbohydrates and transformation and their effects on blood glucose levels follow similar dynamics. These dynamics can be described rather well with a two-compartment model with different uptake and degradation rates [10, 47]. Of course, the complexity of this model can be increased almost freely by adding more compartments and especially by introducing non-linearities or time-dependent flow rates (cf. [47]).

We have used a three-compartment model within the enhanced GP algorithm to create smoothed time series of the original carbohydrate and insulin time series which are subsequently used within the evolved symbolic regression models. The smoothed values are calculated by integrating the following set of differential equations starting with zero values for each compartment $Q_1(0) = Q_2(0) = Q_3(0) = 0$. The input time series X_t is set to the insulin values (I_t) or the carbohydrate values (C_t), respectively. The time series $Q_3(t)$ is used as the result.

$$\frac{dQ_1(t)}{dt} = X_t - \alpha Q_1(t) \tag{13}$$

$$\frac{dQ_2(t)}{dt} = \alpha(Q_1(t) - Q_2(t)) \tag{14}$$

$$\frac{dQ_3(t)}{dt} = \alpha Q_2(t) - \beta Q_3(t) \tag{15}$$

The terminal set only contains four terminal symbols {`SmoothCh`, `SmoothIns`, `LaggedGluc`, `const`}. Real-valued constants are handled exactly in the same way as in tree-based GP. The symbols for `SmoothCh` and `SmoothIns` both contain parameters for the three-compartment model which are initialized randomly (α and β are limited to the range $[0.001 \ldots 0.999]$ and are sampled uniformly from that range) as well as a weight which is initialized by sampling from $N(\mu = 0, \sigma = 10)$. Parameters are manipulated only by mutation operators, whereby the weight is changed by adding a value sampled from $N(\mu = 0, \sigma = 1.0)$ and α and β are changed by adding independently sampled values from $N(\mu = 0, \sigma = 1.0)$. If the resulting value is outside of the interval $[0.001 \ldots 0.999]$ the value is adjusted to the allowed range.

The function set used here is

$$\{+, -, *, /, log(x), exp(x), sin(x), cos(x)\}.$$

Fig. 3 Grammar developed for solving the extraction of models of glycemia for G_{t+30}. It considers the preprocessing of the data described in Section "Data preprocessing" and including some knowledge about the glucose prediction problem

```
# Model expression
<func> ::= <expr>

<expr> ::= (<gl> + <ch> - <ins>) + <cte> - <cte>

# Glucose
<gl> ::= <preop> (<gl>) | <vargl> | <gl> <op> <gl> |
      (-1)*(<gl>)
<vargl> ::= x2|x3|x4|x5|x6|x7|x8

# CH
<ch> ::= <preop> (<ch>) | <varch> | <ch> <op> <ch>
<varch> ::= x9|x10|x11|x12|x13|x14|x15|x16|x17|x18|x19|x20|x21|x22|x23

# Insulin:
<varins> ::= x24|x25|x26|x27|x28|x29|x30|x31|x32|x33|x34|x35|x36|x37|x38
<op> ::= +|-|*|/
<preop> ::= Math.exp|Math.sin|Math.cos|Math.log
<cte> ::= <base>*Math.pow(10,<sign><exponent>)
<base> ::= 1|2|3|4|5|6|7|8|9|10|11|12|13|14|15|16|17|18|19|
 ... 90|91|92|93|94|95|96|97|98|99
<exponent> ::= 1|2|3|4|5|6|8|9
<sign> ::= +|-
```

The GP configuration is similar to the configuration of GP described in the previous section. Again we have not tuned the parameters but used rather robust settings. The initial population of 300 solution candidates was generated randomly using PTC2 [33]. The depth limit for trees was set to 11 levels, the maximum size of trees was set to 100 nodes. The algorithm was run for 100 generations using the same crossover and mutation operators, the mutation probability was set to 15%. We used gender-specific parents selection [45] where one parent is selected using fitness-proportional selection and the other parent is selected randomly. For offspring selection we also used the strict variant as above. We use the identifier GP_spec to refer to this configuration of tree-based genetic programming for symbolic regression in the following sections.

Modeling glycemia by grammatical evolution with data preprocessing and a glucose specific grammar

GE is a branch of GP that emerged twenty years ago as a way to evolve programs using *context free grammars* to generate code [36]. The objective was also to reduce the size of the programs and to minimize, with respect to other approaches, the proportions of non-effective code generated in the decoding process. The most usual way of representing the grammar is in BNF (Backnus-Naur Form) format. We can define a BNF grammar as a 4-tuple $\{S, N, T, Rules\}$, where S is the starting symbol, N is a set of non-terminal symbols, T is the set of terminal symbols and *Rules* are the rules we use to advance in the decoding of the expression, transforming non-terminals in other non-terminals or terminals. The set of rules is responsible for defining the productions of each one of the non-terminal symbols. To complete this process, the definition of a translation operator from the genetic code to the terminals according to the rules is needed. In this case, we select the most usual operator in GE, the modulo operator, symbolized as %. Thus, to know the decoding of a gene we will apply the operation of modulo with the number of productions in the rule for the given symbol:

$$\text{Value} = (\text{Value\_of\_Gene})\%\,\#(\textit{Productions\_for\_the\_symbol})$$

In the case of modeling the glycemia of diabetic patients with GE, the phenotype of an individual is the model expression for prognosis. We have developed a grammar to guide the optimization process of generating suitable models for the training data. We apply a grammar which considers that the prediction for time t may depend on the previous values of glucose, carbohydrates ingestion, and insulin injection, following [26]. In addition, for each temporal horizon we can also use future values of carbohydrates and insulin, up to the limit of the horizon.

Figure 3 shows a grammar we have used to solve the symbolic regression problem that looks for a relation between the glucose in 30 minutes (G_{t+30}) and the input variables. In this case the input variables correspond with the values of the features in Table 2. Here, all the historical data are presented as variables as well as the future values of insulin and meals valid for each prediction horizon. The table shows all the variables with their corresponding historical or future data. Similar as when using other techniques, depending on the prediction horizon we are allowed to use specific sets of features. Again, if we are looking for a model for G_{t+30}, we can use historical data of glucose, carbohydrates and insulin till time t, and also future values of carbohydrates and insulin till $t + 30$. Following the same idea, Table 2 explains the variables permitted for obtaining the models for the $t + 60$, $t + 90$ and $t + 120$ horizons. The aim of the grammar is to generate model expressions by combinations of the input variables, mathematical operators

Table 2 We solve the glucose modeling problem as a symbolic regression problem

Variable	BNF T	Available? (👁)			
y_r	Target	30	60	90	120
Time	x1	👁	👁	👁	👁
$G_{(t)}$	x2	👁	👁	👁	👁
$G_{(t-[120...90[)}$	x3	👁	👁	👁	👁
$G_{(t-[90...60[)}$	x4	👁	👁	👁	👁
$G_{(t-[60...45[)}$	x5	👁	👁	👁	👁
$G_{(t-[45...30[)}$	x6	👁	👁	👁	👁
$G_{(t-[30...15[)}$	x7	👁	👁	👁	👁
$G_{(t-[15...0[)}$	x8	👁	👁	👁	👁
$C_{(t)}$	x9	👁	👁	👁	👁
$C_{(t-[120...90[)}$	x10	👁	👁	👁	👁
$C_{(t-[90...60[)}$	x11	👁	👁	👁	👁
$C_{(t-[60...45[)}$	x12	👁	👁	👁	👁
$C_{(t-[45...30[)}$	x13	👁	👁	👁	👁
$C_{(t-[30...15[)}$	x14	👁	👁	👁	👁
$C_{(t-[15...0[)}$	x15	👁	👁	👁	👁
$C_{(t+]0...15])}$	x16	👁	👁	👁	👁
$C_{(t+]15...30])}$	x17	👁	👁	👁	👁
$C_{(t+]30...45])}$	x18	✗	👁	👁	👁
$C_{(t+]45...60])}$	x19	✗	👁	👁	👁
$C_{(t+]60...75])}$	x20	✗	✗	👁	👁
$C_{(t+]75...90])}$	x21	✗	✗	👁	👁
$C_{(t+]90...105])}$	x22	✗	✗	✗	👁
$C_{(t+]105...120])}$	x23	✗	✗	✗	👁
$I_{(t)}$	x24	👁	👁	👁	👁
$I_{(t-[120...90[)}$	x25	👁	👁	👁	👁
$I_{(t-[90...60[)}$	x26	👁	👁	👁	👁
$I_{(t-[60...45[)}$	x27	👁	👁	👁	👁
$I_{(t-[45...30[)}$	x28	👁	👁	👁	👁
$I_{(t-[30...15[)}$	x29	👁	👁	👁	👁
$I_{(t-[15...0[)}$	x30	👁	👁	👁	👁
$I_{(t+]0...15])}$	x31	👁	👁	👁	👁
$I_{(t+]15...30])}$	x32	👁	👁	👁	👁
$I_{(t+]30...45])}$	x33	✗	👁	👁	👁
$I_{(t+]45...60])}$	x34	✗	👁	👁	👁
$I_{(t+]60...75])}$	x35	✗	✗	👁	👁
$I_{(t+]75...90])}$	x36	✗	✗	👁	👁
$I_{(t+]90...105])}$	x37	✗	✗	✗	👁
$I_{(t+]105...120])}$	x38	✗	✗	✗	👁

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

$\{+, -, *, /\}$ and functions exp, sin, cos and log. We refer the interested reader to [26] to see an extended example of the GE decoding process for a glucose modeling problem.

The use of grammars allows directing the search by providing knowledge to the optimization algorithm. In contrast, a poor grammar design can also lead to problems in reaching some parts of the search space. In order to apply GE we need to perform the same data preprocessing as previously described, and we will try to obtain an expression according to the variables available at each moment. As in previous works [15, 26, 44], the first set of grammars is used to direct the search introducing some knowledge on it. Thus, the start symbol is always transformed into an expression in the form of:

$$< expr > ::= < gl > + < ch > - < ins > \quad (16)$$

which indicates that the final expression *(phenotype)* is a combination of variables of glucose, $< gl >$, with variables of carbohydrates, $< ch >$, that have a positive impact, and variables of insulin, $< ins >$, that have a negative impact.[4] The grammar supports the idea that meals (carbohydrates) always increment the glucose value while insulins always reduce the glucose values. We have implemented the GE process in Java using the ABSys JECO library [1] using compilable phenotypes to speed up the evaluation of individuals [13]. Again, variables in Table 2 are obtained by the average indicated in Eq. 1.

The rest of the grammar provides rules for decoding the individuals and include the four basic arithmetic operations (rule for `<op>`), the mentioned mathematical functions (rule for `<preop>`), and a way to generate constants through the non-terminal symbols `<cte>`, `<base>`,`<sign>` and `<exponent>`. For the models of G_{t+60}, G_{t+90} and G_{t+120}, we produced similar grammars using the available variables for each time horizon (see Table 2). RSME was used as fitness function. We use standard genetic operators as in a typical genetic algorithm, with a population size of 200 individuals, one-point crossover with 0.85 probability and individual mutation probability of 0.25. The length of the chromosome is 300 codons, the maximum number of wrappings is 5.[5] and we run 250 generations of the algorithm 5 times on each cross validation fold. We will use the identifier GE_spec to refer to this configuration of grammatical evolution for symbolic regression in the following sections.

Modeling glycemia by grammatical evolution as a direct symbolic regression problem with a simpler grammar

In the case of GE, we have also made an additional implementation to the one described in the previous section. The alternative implementation follows the idea of the one realized in GP (described in Section "Symbolic regression by tree-based genetic programming"), although with some modifications. In this case a more restrictive implementation has been made in terms of reducing the number of features or variables to be used, as well as functions and operators. This approach has been investigated by other

[4]The expression in Fig. 3 includes also some constants ($< cte >$) not present in Eq. 16 for the sake of simplicity of the explanation.

[5]We use wrapping of 5 because with lower values we obtain high percentages of non valid solutions during the initial generations.

```
<func> ::= <expr>

<expr> ::= (<expr> <op> <expr>) | (<cte> <op> <expr>)  |
      <var>

<var> ::= <varch> | <varins> | <vargl>

<op> ::= +|-|*

<vargl> ::=  x2 | x4 | x7

<varch> ::= x9 | (x10 + x11) | (x12 + x13) |
      (x14 + x15) | (x16 + x17) | (x18 + x19) |
      (x20 + x21) | (x22 + x23)

<varins> ::= x24 | (x25 + x26) | (x27 + x28) |
      (x29  + x30) | (x31  + x32) | (x33  + x34) |
      (x35  + x36) | (x37  + x38)

<cte> ::=  <factor> * <digit>
<factor> ::= 0.1 | 0.01 | 0.001| 0.0001|  1
<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10
```

Fig. 4 Grammar developed for solving the extraction of models of glycemia for G_{t+120}. It considers the preprocessing of the data described in Section "Data preprocessing", and the reduction in the number of variables

authors in both GE [19, 37], where authors developed and work with different grammars, and GP [34] where the authors reduced the number of features and still retrieved appealing results. The reduction of variables has been done after observing that the information of two consecutive values of preprocessed glucose values, for instance $G_{(t-[15...0[)}$ and $G_{(t-[30...15[)}$ are very similar and therefore do not provide any special information to use several of them. In the case of carbohydrates and insulin instead of eliminating variables we add two new ones, since we want to preserve all the inputs. This second approach also reduces the number of operators to the three basic arithmetic operations $+$, $-$ and $*$. In most of the phenotypes, the division operation only incorporates complexity without being part of the best solutions. We have also reduced the complexity of the rules to generate constants for two reasons. First, since we have reduced the number of rules for the rest of the terminals, it is not convenient for the constants to consume a large number of genes on the chromosome, since it could lead to an increase in the number of wrappings, or the number of non-feasible solutions. Second, during the tuning of the algorithm in the preliminary experiments we have observed that good solutions do not incorporate large constants. Hence, those complex constants can be simplified.

Figure 4 shows the grammar used in this second approach for obtaining models for G_{t+120}. For the models of G_{t+30},G_{t+60} and G_{t+90}, we produced similar grammars using the available variables for each time horizon, which can be seen on Table 3. We used a similar configuration to the previous one, with RSME as objective function, a genetic algorithm with typical operators, 200 individuals, one point crossover with 0.85 probability, individual mutation probability of 0.25, 300 codons, 5 wrappings and 250 generations. We used the identifier GE_gen to refer to this configuration of grammatical evolution for symbolic regression in the following sections.

Experimental results

Ten T1DM patients (n = 10) have been selected for the observational study, based on conditions of good glucose control. Data from patients were acquired over multiple weeks using Medtronic CGMs. Log entries were stored in five-minute intervals. able shows the number of data points available for each patient. In this dataset we have at least

Table 3 The table shows the variables with their corresponding historical or future data for the second approach GE_gen, where we reduced the number of features

Variable	BNF T	Available? (👁)			
y_r	Target	30	60	90	120
$G_{(t)}$	x2	👁	👁	👁	👁
$G_{(t-[120...90[)}$	x3	✗	👁	✗	👁
$G_{(t-[90...60[)}$	x4	👁	✗	👁	✗
$G_{(t-[60...45[)}$	x5	✗	👁	✗	👁
$G_{(t-[45...30[)}$	x6	👁	✗	👁	✗
$G_{(t-[30...15[)}$	x7	✗	👁	✗	👁
$G_{(t-[15...0[)}$	x8	👁	👁	👁	👁
$C_{(t)}$	x9	👁	👁	👁	👁
$I_{(t)}$	x24	👁	👁	👁	👁

The reduction of variables has been done after observing that the information of two consecutive values of preprocessed glucose values are very similar. In the case of carbohydrates and insulin variables are the same of Table 2, although we add them in pairs. For each horizon an (👁) indicates that the variable is available for prediction, while a (✗) indicates that it is not permitted

Table 4 Descriptive statistics for the data sets for all ten patients

Patient	Average (mg/dl)	Std. Deviation (mg/dl)	Time glucose <70 mg/dl	Time glucose >250 mg/dl	Time in target [70-180] mg/dl
1	157.67	62.25	4.48%	7.31%	37.57%
2	145.44	64.02	8.33%	6.18%	41.83%
3	143.46	45.45	2.14%	2.26%	49.18%
4	150.47	56.70	4.12%	5.14%	41.29%
5	139.17	67.93	14.17%	7.10%	41.89%
6	142.58	60.08	10.08%	4.44%	41.21%
7	176.33	68.35	3.65%	14.28%	27.07%
8	135.34	46.11	4.85%	1.84%	54.06%
9	146.82	59.82	7.54%	5.20%	39.49%
10	166.13	86.12	9.33%	16.16%	35.69%

Values in range 180-250 are $(100\% - \text{Time glucose}_{<70} - \text{Time glucose}_{>250} - \text{Time in target}_{70-180})$

10 complete days of data for each patient. These days are not necessarily consecutive neither the same days for the patients. Each log entry contains the date and time, the blood glucose value, the amount of insulin (injected via pump), and the amount of carbohydrate intake as estimated by the patients. The population characterization is female (80%), average age 42.3 (+/−11.07), years of disease 27.2 (+/ − 10.32), years with pump therapy 10 (+/ − 4.98), weight 64.78 (+/ − 13.31) kg, HbA1c average of 7.27% (+/−0.5%). The average number of days with data is 44.80 (+/ − 30.73). Table 4 shows some statistics about the glucose levels of the data acquired from the patients and Table 5 the number of data points for each patient.

The Clarke error grid (CEG) approach is commonly used to test the clinical significance of differences between a glucose measurement technique and the venous blood glucose reference measurements [9]. In analogy, Clarke errors can be used to show the differences between predicted values and reference (actual) values. CEG represent the values of the prediction versus the reference (actual) values on a cartesian diagram. Thus, the point (x, y) in the XY cartesian graphic represents that when the actual value was x, the predicted value was y, being $Y = X$ the perfect prediction. The particularity of CEG is that the graphic is that the XY graph is divided into a grid of zones depending on the severity of the misspredicted. There are five zones delimited in the graph (A to E), with the following meanings:

- Zones A and B: values on Zone A represents the glucose values that deviate from the reference values by 20% or less and those that are in the hypoglycemic range (<70 mg/dl), not only the predicted value but also the reference value. Predicted values on zone B deviate from the reference values by more than 20% although the clinical treatment will be correct with a high probability. The values that fall within zones A and B are clinically exact and/or acceptable and thus the clinical treatment will be correct. We will treat those classes as a combined category in the analysis of our experimental results.
- Zone C: The values in this zone could be dangerous in some situations. The goal is to minimize predictions in this category.
- Zones D-E: The values included in those areas are potentially dangerous, since the prediction is far from being acceptable and the indicated treatment will be different from the correct. Again, the goal is to minimize predictions in this category.

Table 5 Table shows the number of data points available for each patient

Pat. #	1	2	3	4	5	6	7	8	9	10
Data pts	4018	23534	23821	20090	8036	12628	6888	6027	5740	4305

The data were collected from a system formed by a continuous glucose monitor (CGM) and an insulin pump attached to the patient's body. Measures of both variables were taken each 5 minutes, and carbohydrates ingestions were also annotated and collected. In this dataset we have at least 10 complete days of data for each patient. These days are not necessarily consecutive neither the same days for the patients

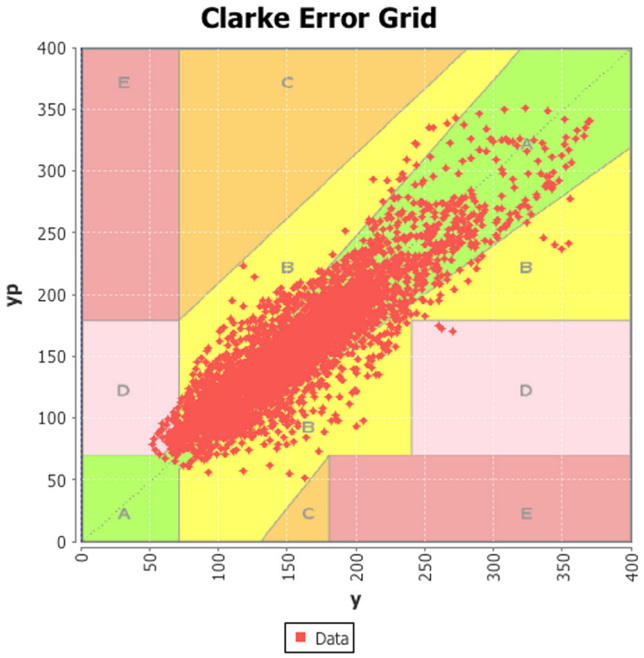

Fig. 5 Clarke error grid for Patient 1- Horizon 30 minutes predictions

In 2000 Parkes et al. [39] revisited the definition of the zone and constructed a set of new error grids by consulting a large panel of clinicians. Currently, in the field of endocrinology there is no general consensus for evaluating errors in the measurement of blood glucose for diabetics, so we use here the original CEG to analyze the results. Figure 5 shows the Clarke error grid for Patient 1- Horizon 30 minutes predictions.

In this experiment we have applied to the patients' dataset all the previously described algorithms: GE_gen, GE_spec, GP_spec, GP, KNN and RF. Additionally we have considered two baseline predictors: The first one considers the average glucose of the previous values in the past two hours; we denote it here as Avg. The second baseline considers as prediction the last known value of the glucose; it is here denoted as Last.

In the following we describe the results of the experiments and analyze the performance of all the predictors.

Figures 6, 7, 8, and 9 show the results for all tested methods aggregated over the ten patients. We only show plots for the two-hour ($t + 120$) prediction models. Each box plot shows the fractions of predictions for one of the five Clarke Error Grid categories. The full data for the all prediction horizons are given in Tables 6 and 7.

In Tables 6 and 7 we highlight the modeling result with the highest fraction of predictions with a Clarke error of A or B. In general, the prediction accuracy is better for the short forecasting horizons and gradually become worse as the forecasting horizon is increased from 30 minutes to two hours. In 18 out of 32 cases one of the GE variants performed best, while in eleven cases one of the tree-based GP variants performed best.

To analyse which one of the tested methods performs best over all patients we calculated rankings ($1 \ldots 8$) for all datasets and all time offsets and calculated the average rank for each method. These ranking values are shown in Table 8. As suggested in [20] we used the non-parametric Friedman test to check the null-hypothesis that all methods are ranked equally. The data in Table 8 leads us to reject the null-hypothesis for all four forecast horizons ($\alpha = 0.05$). For pairwise comparison of classifiers we use the Nemenyi test [35] as post-hoc analysis with a critical value $q_{\alpha=0.05} = 3.031$ with $k = 8$ classifiers [20]. The resulting critical difference for ranks is 3.32. Figures 10, 11, 12, 13 show the critical difference diagrams whereby methods with average ranks closer than the critical difference are connected to groups.

It is important to note that using the last known value for the prediction also produces reasonable results for the shorter forecast horizons of 30 minutes and 60 minutes (see Figs. 10 and 11). In particular, the available data does not allow us to show that the best methods achieve significantly better rankings than the last-value baseline. For the 90 minutes and the 120 minutes forecasts GP_gen, GP_spec, RF

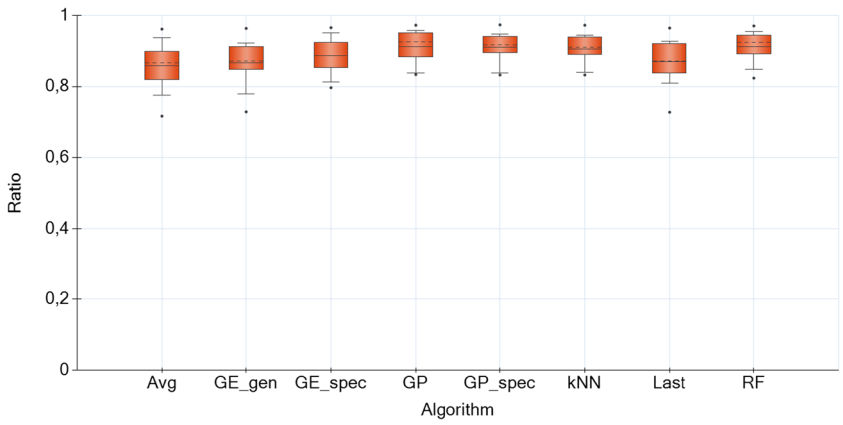

Fig. 6 Box plot of fractions of predictions in Clarke error categories A and B. These are categories of correct predictions. Higher values are better. All learning algorithms produce similar results. RF, GP variants and KNN produce the best results

Table 6 Fractions of predictions (in percent) on independent test data

	t+30				t+60			
Alg	A + B	C	D	E	A + B	C	D	E
Patient1								
Avg	42.0 + 48.0	0.0	10.1	0.0	41.8 + 48.0	0.0	10.1	0.0
Last	77.2 + 21.7	0.1	1.0	0.0	57.8 + 39.0	0.8	2.1	0.2
kNN	81.1 + 17.9	0.1	1.0	0.0	64.3 + 33.2	0.3	2.1	0.1
RF	**85.7 + 13.6**	0.0	0.7	0.0	**68.8 + 29.4**	0.0	1.7	0.0
GE_spec	**83.6 + 15.7**	0.2	0.5	0.0	64.8 + 32.6	1.0	1.4	0.2
GE_gen	**85.7 + 13.6**	0.3	0.4	0.0	65.4 + 32.0	0.7	1.7	0.2
GP	84.4 + 14.3	0.0	1.3	0.0	66.6 + 31.1	0.1	2.2	0.0
GP_spec	84.6 + 14.0	0.0	1.4	0.0	68.2 + 29.5	0.0	2.4	0.0
Patient2								
Avg	38.4 + 47.9	0.0	13.7	0.0	38.5 + 47.8	0.0	13.7	0.0
Last	72.8 + 24.8	0.1	2.3	0.0	52.4 + 41.8	0.9	4.8	0.1
kNN	85.6 + 12.4	0.0	1.9	0.0	64.1 + 31.6	0.1	4.2	0.0
RF	91.5 + 7.0	0.0	1.5	0.0	**73.7 + 22.8**	0.1	3.4	0.0
GE_spec	80.6 + 17.8	0.1	1.4	0.0	56.2 + 38.8	1.2	3.8	0.1
GE_gen	85.5 + 13.0	0.4	1.1	0.0	59.1 + 36.5	0.8	3.5	0.1
GP	**91.8 + 6.8**	0.0	1.3	0.0	71.6 + 24.2	0.1	4.1	0.0
GP_spec	86.7 + 11.1	0.0	2.2	0.0	68.0 + 27.3	0.1	4.5	0.0
Patient3								
Avg	51.3 + 44.7	0.0	3.9	0.0	51.3 + 44.9	0.0	3.8	0.0
Last	82.0 + 17.2	0.0	0.8	0.0	65.7 + 32.7	0.1	1.5	0.0
kNN	85.6 + 13.5	0.0	0.9	0.0	71.0 + 27.5	0.0	1.5	0.0
RF	89.4 + 9.7	0.0	1.0	0.0	75.5 + 22.8	0.0	1.6	0.0
GE_spec	**85.9 + 13.5**	0.0	0.6	0.0	**68.8 + 29.8**	0.1	1.3	0.0
GE_gen	87.1 + 12.2	0.2	0.5	0.0	70.0 + 28.2	0.4	1.4	0.0
GP	89.1 + 10.1	0.0	0.9	0.0	73.7 + 24.6	0.0	1.6	0.0
GP_spec	87.6 + 11.2	0.0	1.2	0.0	73.6 + 24.7	0.0	1.8	0.0
Patient4								
Avg	41.1 + 48.2	0.0	10.7	0.0	41.0 + 48.4	0.0	10.6	0.0
Last	76.4 + 22.1	0.0	1.4	0.0	57.6 + 38.9	0.5	2.9	0.1
kNN	82.2 + 16.3	0.0	1.5	0.0	64.1 + 32.9	0.2	2.9	0.0
RF	86.6 + 11.5	0.0	1.9	0.0	68.1 + 29.0	0.1	2.8	0.0
GE_spec	81.8 + 17.1	0.1	0.9	0.0	60.7 + 36.3	0.5	2.4	0.0
GE_gen	**85.0 + 14.1**	0.3	0.6	0.0	**62.8 + 34.4**	0.5	2.4	0.0
GP	86.2 + 12.2	0.0	1.6	0.0	66.0 + 31.1	0.1	2.8	0.0
GP_spec	83.1 + 15.0	0.0	1.9	0.0	65.8 + 31.0	0.1	3.1	0.0
Patient5								
Avg	33.3 + 45.2	0.0	21.5	0.0	33.3 + 45.1	0.0	21.6	0.0
Last	66.5 + 27.7	0.3	5.4	0.1	49.2 + 41.0	1.0	8.1	0.8
kNN	72.1 + 21.8	0.1	6.0	0.0	52.4 + 37.4	0.6	9.3	0.3
RF	76.5 + 17.5	0.0	6.0	0.0	58.6 + 33.0	0.3	7.9	0.2
GE_spec	72.0 + 23.2	0.9	3.8	0.1	**51.8 + 40.0**	0.8	6.8	0.5
GE_gen	**75.7 + 20.9**	0.3	t.1	0.1	50.9 + 37.8	3.0	7.8	0.5

Table 6 (continued)

	t+30				t+60			
Alg	A + B	C	D	E	A + B	C	D	E
GP	75.9 + 17.4	0.0	6.6	0.1	54.8 + 33.8	0.5	10.7	0.2
GP_spec	74.6 + 18.8	0.0	6.6	0.0	54.0 + 34.3	0.1	11.6	0.0
Patient6								
Avg	36.5 + 50.0	0.0	13.5	0.0	36.5 + 49.9	0.0	13.6	0.0
Last	69.5 + 26.9	0.1	3.4	0.0	52.4 + 40.4	1.3	5.5	0.5
kNN	74.6 + 21.9	0.0	3.4	0.0	56.3 + 36.8	0.4	6.3	0.2
RF	79.9 + 16.5	0.0	3.6	0.0	61.1 + 31.8	0.1	7.1	0.0
GE_spec	75.4 + 21.8	0.4	2.4	0.0	54.8 + 38.0	1.5	5.4	0.3
GE_gen	**77.3 + 20.0**	0.5	2.1	0.0	**56.4 + 37.3**	1.0	5.0	0.3
GP	80.3 + 16.3	0.0	3.4	0.0	61.2 + 31.5	0.2	7.0	0.1
GP_spec	77.5 + 18.9	0.0	3.6	0.0	61.1 + 30.8	0.2	7.9	0.0
Patient7								
Avg	40.6 + 39.4	0.0	20.0	0.0	40.8 + 39.3	0.0	19.9	0.0
Last	69.9 + 27.2	0.3	2.6	0.0	54.5 + 37.9	2.0	5.0	0.7
kNN	76.0 + 21.1	0.2	2.7	0.0	60.9 + 33.0	1.3	4.6	0.1
RF	79.9 + 17.1	0.1	2.9	0.0	**64.6 + 30.5**	0.5	4.5	0.0
GE_spec	74.3 + 23.1	0.4	2.2	0.0	57.6 + 34.8	2.1	5.0	0.5
GE_gen	**78.1 + 19.9**	0.6	1.4	0.0	59.2 + 34.3	1.5	4.6	0.5
GP	79.1 + 18.3	0.3	2.3	0.0	60.9 + 33.7	0.7	4.6	0.0
GP_spec	77.5 + 19.4	0.2	2.9	0.0	64.5 + 30.4	0.7	4.4	0.0
Patient8								
Avg	46.6 + 46.9	0.0	6.5	0.0	46.9 + 46.9	0.0	6.3	0.0
Last	69.5 + 27.7	0.3	2.4	0.0	49.7 + 45.3	0.9	3.8	0.3
kNN	72.5 + 24.1	0.1	3.2	0.0	55.7 + 38.8	0.4	5.0	0.1
RF	77.7 + 18.1	0.0	4.2	0.0	61.7 + 32.4	0.0	5.9	0.0
GE_spec	74.2 + 23.5	0.2	2.1	0.0	52.6 + 43.1	0.4	3.7	0.2
GE_gen	**74.7 + 23.5**	0.1	1.8	0.0	**53.4 + 42.6**	0.5	3.3	0.1
GP	77.3 + 19.4	0.1	3.3	0.0	58.3 + 36.1	0.1	5.5	0.0
GP_spec	77.1 + 19.6	0.0	3.3	0.0	63.8 + 31.0	0.0	5.1	0.0
Patient9								
Avg	40.9 + 46.1	0.0	13.0	0.0	40.7 + 46.4	0.0	12.9	0.0
Last	72.1 + 24.6	0.2	3.0	0.0	57.1 + 36.2	1.4	4.9	0.4
kNN	75.7 + 21.1	0.0	3.3	0.0	57.1 + 36.6	0.3	5.8	0.1
RF	77.6 + 17.3	0.0	5.0	0.0	61.2 + 31.8	0.2	6.8	0.0
GE_spec	77.2 + 20.5	0.0	2.2	0.0	58.7 + 34.9	1.9	4.3	0.3
GE_gen	**80.8 + 17.4**	0.3	1.5	0.0	**59.5 + 34.4**	1.3	4.6	0.2
GP	82.4 + 14.9	0.0	2.7	0.0	62.1 + 31.2	0.1	6.6	0.1
GP_spec	79.2 + 17.1	0.0	3.7	0.0	26.0 + 61.1	6.3	6.0	0.6
Patient10								
Avg	29.1 + 42.1	0.0	28.8	0.0	28.9 + 42.5	0.0	28.6	0.0
Last	62.3 + 32.4	0.8	4.5	0.0	42.1 + 45.2	3.0	9.3	0.4
kNN	73.7 + 22.4	0.2	3.7	0.0	**52.6 + 38.2**	1.3	7.7	0.2

Table 6 (continued)

	t+30				t+60			
Alg	A + B	C	D	E	A + B	C	D	E
RF	77.4 + 18.3	0.1	4.1	0.0	52.9 + 37.9	0.7	8.5	0.1
GE_spec	74.0 + 23.1	0.3	2.6	0.0	46.1 + 43.4	2.4	8.0	0.2
GE_gen	76.8 + 20.5	0.8	1.8	0.0	49.8 + 40.5	3.1	6.4	0.2
GP	**83.2 + 14.8**	0.1	1.9	0.0	57.3 + 33.3	0.8	8.6	0.0
GP_spec	76.2 + 20.1	0.1	3.6	0.0	54.7 + 35.6	0.6	9.1	0.1

For each patients and prediction horizon, the best modeling results are highlighted

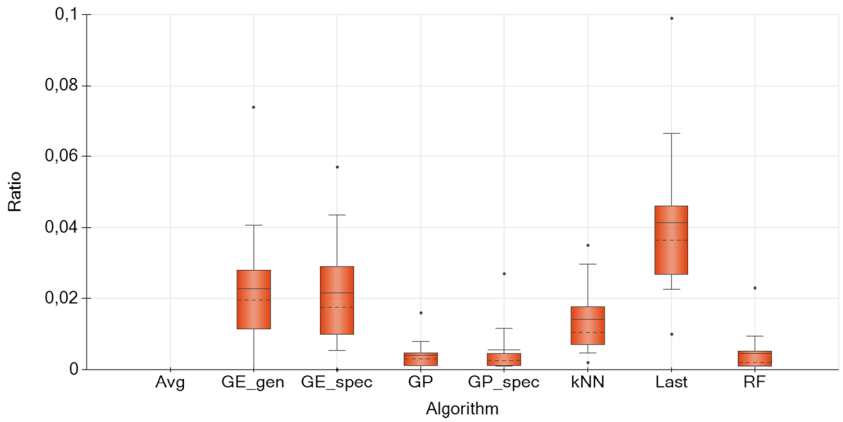

Fig. 7 Box plot of fractions of predictions in Clarke error category C. Lower values are better. The baseline algorithm Avg produces no errors in this category and is therefore best. GP variants and RF also produce good results

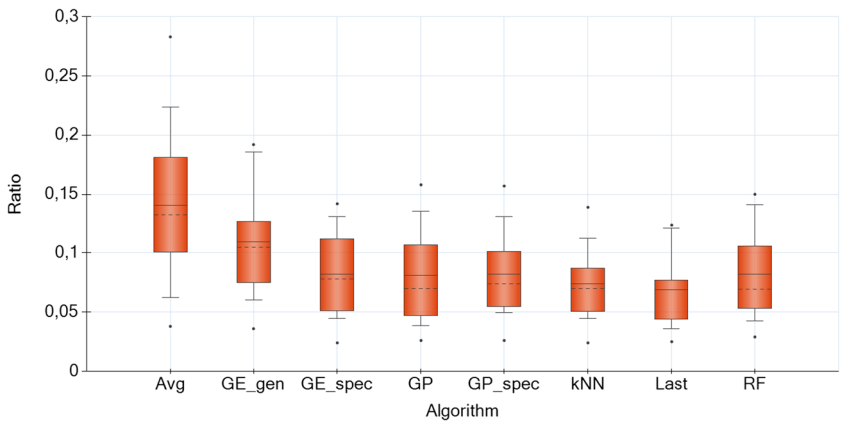

Fig. 8 Box plot of fractions of predictions in Clarke error category D. This is a category of serious errors. Lower values are better. The naive last known value baseline performs best

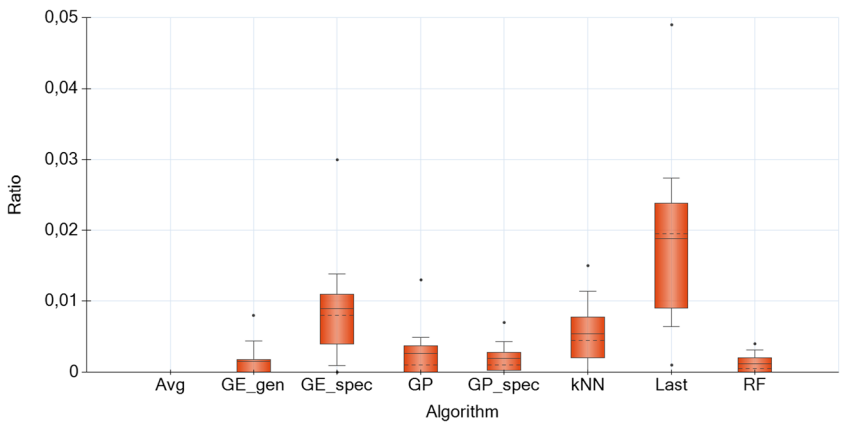

Fig. 9 Box plot of fractions of predictions in Clarke error category E. This is the category of critical errors. Lower values are better. The baseline Avg produces no errors in this category and is therefore best. GE_gen, GP, and RF all produce less than 1% of serious errors

Table 7 Fractions of predictions (in percent) on independent test data

	t+90				t+120			
Alg	A + B	C	D	E	A + B	C	D	E
Patient1								
Avg	41.6 + 48.4	0.0	10.0	0.0	41.6 + 48.5	0.0	9.9	0.0
Last	47.4 + 46.3	2.5	3.2	0.5	42.6 + 49.6	3.2	3.7	0.8
kNN	55.5 + 40.4	0.6	3.4	0.1	49.5 + 44.6	1.0	4.8	0.0
RF	58.9 + 37.7	0.3	3.1	0.0	53.5 + 41.8	0.3	4.4	0.0
GE_spec	53.6 + 42.0	0.9	3.3	0.2	50.1 + 41.7	1.1	7.2	0.0
GE_gen	53.3 + 42.1	1.2	3.2	0.2	48.2 + 44.4	2.3	4.7	0.4
GP	60.0 + 36.7	0.1	3.2	0.0	**54.2 + 41.3**	0.5	4.0	0.0
GP_spec	**60.3 + 36.6**	0.0	3.1	0.0	52.5 + 41.9	0.2	5.2	0.1
Patient2								
Avg	38.6 + 47.7	0.0	13.7	0.0	38.6 + 47.8	0.0	13.6	0.0
Last	41.8 + 48.5	2.3	6.5	0.9	35.7 + 49.6	4.8	7.6	2.3
kNN	53.0 + 40.8	0.3	5.7	0.1	46.1 + 46.2	0.6	6.9	0.2
RF	**62.9 + 32.5**	0.1	4.5	0.0	55.4 + 38.6	0.1	5.9	0.0
GE_spec	44.6 + 45.9	1.6	7.5	0.4	39.8 + 47.6	1.7	10.9	0.0
GE_gen	45.6 + 45.0	3.2	5.7	0.5	40.0 + 48.5	1.3	9.3	0.8
GP	59.5 + 35.0	0.2	5.3	0.0	**64.6 + 30.5**	0.2	4.6	0.0
GP_spec	57.2 + 36.5	0.2	6.1	0.0	49.1 + 43.2	0.3	7.4	0.1
Patient3								
Avg	51.3 + 44.9	0.0	3.8	0.0	51.3 + 44.9	0.0	3.8	0.0
Last	56.3 + 41.1	0.5	2.1	0.1	50.6 + 45.9	1.0	2.5	0.1
kNN	62.8 + 35.0	0.2	2.0	0.0	**57.7 + 39.7**	0.2	2.4	0.0
RF	67.3 + 30.4	0.0	2.3	0.0	63.2 + 33.9	0.0	2.9	0.0
GE_spec	58.5 + 39.0	0.4	2.0	0.0	52.5 + 43.9	0.0	3.6	0.0
GE_gen	59.8 + 37.5	0.6	2.1	0.0	53.4 + 43.2	0.9	2.4	0.1
GP	**64.4 + 33.4**	0.0	2.2	0.0	**60.2 + 37.2**	0.0	2.6	0.0
GP_spec	66.6 + 31.2	0.0	2.2	0.0	**61.8 + 35.6**	0.1	2.6	0.0
Patient4								
Avg	40.9 + 48.4	0.0	10.6	0.0	40.9 + 48.4	0.0	10.6	0.0
Last	48.0 + 46.2	1.5	3.9	0.4	42.3 + 50.0	2.5	4.5	0.7
kNN	53.6 + 41.8	0.6	4.0	0.1	48.1 + 46.0	1.0	4.7	0.2
RF	**57.4 + 38.4**	0.2	4.0	0.0	**50.8 + 43.7**	0.3	5.1	0.0
GE_spec	50.4 + 44.2	1.2	3.9	0.3	43.1 + 47.2	1.3	8.4	0.1
GE_gen	50.9 + 43.3	1.6	3.8	0.3	44.4 + 47.7	2.0	5.4	0.4
GP	54.9 + 40.7	0.3	4.2	0.0	49.2 + 44.9	0.3	5.6	0.0
GP_spec	57.1 + 38.4	0.1	4.4	0.0	49.5 + 44.7	0.3	5.4	0.1
Patient5								
Avg	33.3 + 45.0	0.0	21.7	0.0	33.1 + 45.2	0.0	21.7	0.0
Last	39.0 + 46.8	2.7	10.3	1.3	32.8 + 49.2	4.0	12.1	2.0
kNN	43.1 + 43.7	1.4	11.3	0.5	37.7 + 45.6	1.9	13.9	0.8
RF	**49.0 + 40.0**	0.5	10.3	0.3	**42.4 + 42.7**	0.6	14.0	0.3
GE_spec	39.1 + 46.3	2.9	11.1	0.7	35.8 + 42.7	2.9	18.5	0.0
GE_gen	41.7 + 44.2	3.5	9.7	0.9	35.7 + 48.7	1.5	13.0	1.1

Table 7 (continued)

	t+90				t+120			
Alg	A + B	C	D	E	A + B	C	D	E
GP	44.2 + 43.1	0.4	12.0	0.2	40.4 + 42.9	0.4	15.8	0.4
GP_spec	45.8 + 40.1	0.2	13.8	0.0	40.6 + 42.7	0.5	15.7	0.4
Patient6								
Avg	36.6 + 49.7	0.0	13.7	0.0	36.6 + 49.5	0.0	13.9	0.0
Last	43.4 + 45.6	2.7	6.8	1.5	38.7 + 47.3	3.8	7.7	2.4
kNN	48.5 + 42.3	0.8	8.1	0.4	**44.3 + 45.0**	1.1	8.9	0.7
RF	52.5 + 38.1	0.3	9.1	0.1	46.7 + 42.3	0.1	10.7	0.2
GE_spec	42.7 + 45.5	1.3	10.3	0.1	40.3 + 46.8	0.0	12.9	0.0
GE_gen	**45.5 + 45.4**	1.5	6.7	0.9	39.1 + 49.7	0.6	9.8	0.8
GP	51.2 + 38.9	0.3	9.4	0.1	46.5 + 42.4	0.1	10.7	0.3
GP_spec	54.7 + 35.3	0.2	9.6	0.1	49.3 + 39.9	0.2	10.3	0.2
Patient7								
Avg	41.1 + 39.1	0.0	19.8	0.0	41.3 + 39.2	0.0	19.5	0.0
Last	44.5 + 42.9	4.1	6.6	1.8	38.3 + 45.1	6.3	7.7	2.5
kNN	53.2 + 38.2	2.1	5.8	0.6	46.6 + 42.4	2.9	7.1	1.1
RF	57.2 + 35.8	0.7	6.1	0.2	**51.5 + 39.9**	0.8	7.5	0.2
GE_spec	45.4 + 43.9	2.4	7.2	1.1	41.8 + 42.7	2.5	12.2	0.8
GE_gen	46.3 + 37.3	5.7	9.7	1.0	38.4 + 43.0	5.7	11.7	1.2
GP	53.1 + 39.2	0.8	6.6	0.3	46.8 + 41.3	0.7	10.7	0.4
GP_spec	**57.7 + 36.4**	0.6	5.2	0.2	50.8 + 40.5	1.0	7.4	0.3
Patient8								
Avg	47.4 + 46.1	0.0	6.4	0.0	47.9 + 45.5	0.0	6.5	0.0
Last	40.5 + 52.6	1.9	4.5	0.6	36.6 + 55.4	2.4	4.4	1.2
kNN	48.5 + 45.2	0.4	5.8	0.1	44.5 + 48.9	0.5	5.9	0.2
RF	56.0 + 37.7	0.0	6.4	0.0	53.9 + 39.6	0.0	6.4	0.0
GE_spec	49.5 + 42.8	1.2	6.5	0.1	48.0 + 43.5	2.2	6.3	0.0
GE_gen	**44.2 + 51.1**	0.2	4.2	0.3	**43.3 + 51.7**	0.0	5.0	0.0
GP	54.5 + 39.2	0.0	6.2	0.0	**55.5 + 39.5**	0.0	5.0	0.0
GP_spec	57.9 + 36.3	0.0	5.8	0.0	55.8 + 38.3	0.1	5.7	0.0
Patient9								
Avg	40.4 + 46.7	0.0	12.9	0.0	40.2 + 46.9	0.0	12.9	0.0
Last	49.8 + 40.7	2.6	5.8	1.2	45.2 + 43.2	3.5	6.2	1.9
kNN	49.3 + 42.6	0.8	6.9	0.4	43.6 + 46.1	1.4	8.1	0.7
RF	53.6 + 37.1	0.3	8.9	0.1	49.6 + 40.1	0.1	10.1	0.1
GE_spec	**50.5 + 42.0**	1.6	5.1	0.8	42.3 + 43.8	3.7	10.0	0.2
GE_gen	50.3 + 41.7	1.4	6.2	0.5	45.1 + 43.3	4.2	6.3	1.1
GP	54.9 + 37.1	0.2	7.7	0.1	**49.3 + 41.8**	0.3	8.4	0.2
GP_spec	55.4 + 36.4	0.0	8.0	0.1	49.7 + 40.7	0.1	9.5	0.0
Patient10								
Avg	28.8 + 42.9	0.0	28.3	0.0	28.8 + 42.9	0.0	28.3	0.0
Last	33.4 + 44.9	7.6	11.9	2.2	28.5 + 44.3	9.9	12.4	4.9
kNN	41.9 + 45.3	2.5	9.4	0.8	**36.9 + 47.1**	3.5	11.0	1.5
RF	41.4 + 45.9	1.1	11.3	0.4	38.8 + 43.5	2.3	15.0	0.4
GE_spec	34.8 + 46.9	6.3	10.5	1.5	28.1 + 44.8	7.4	19.2	0.4
GE_gen	38.2 + 46.2	3.8	10.1	1.6	29.8 + 49.9	3.1	14.2	3.0
GP	43.2 + 43.3	1.8	11.3	0.3	35.9 + 47.9	1.6	13.3	1.3
GP_spec	**43.3 + 44.1**	0.9	11.4	0.3	37.4 + 46.4	2.7	12.8	0.7

For each patients and prediction horizon, the best modeling results are highlighted

Table 8 Average rankings for all methods and time offsets based on the fraction of correctly predicted cases (A+B)

	t+30	t+60	t+90	t+120
Avg	8.00	7.95	7.80	7.45
Last	4.60	5.20	6.50	6.45
kNN	4.75	3.15	3.35	3.05
RF	5.20	3.10	2.70	2.55
GE_spec	2.20	4.00	5.50	6.70
GE_gen	1.45	3.55	4.60	4.70
GP	3.75	4.05	2.65	2.55
GP_spec	6.00	5.00	2.90	2.55

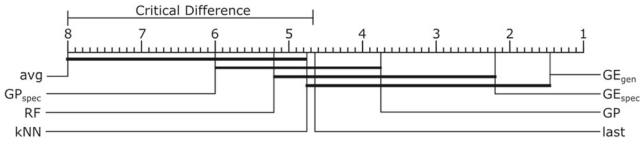

Fig. 10 Average ranking of methods for 30 minute predictions. The critical difference analysis shows that the average rankings of the two GE variants are significantly better than Avg, GP_spec, and RF

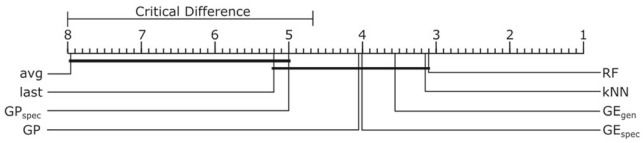

Fig. 11 Average ranking of methods for 60 minute predictions. RF, kNN, GE_gen, GE_spec, and GP are significantly better than the average value prediction (Avg). For the last value prediction (Last) and GP_spec it is not possible to make a statement to which of the two groups they belong given the data

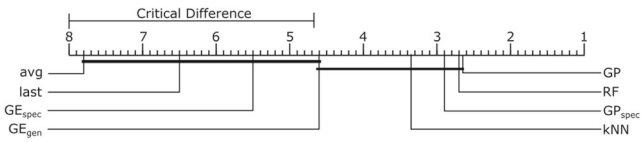

Fig. 12 Average ranking of methods for 90 minute predictions. GP, RF, GP_spec and KNN are significantly better than Avg, Last, and GE_spec. For GE_gen the available data does not allow us to assign it to one of the two groups

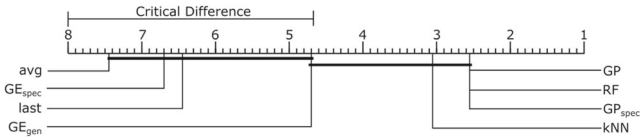

Fig. 13 Average ranking of methods for 120 minute predictions. GP, RF, GP_spec and KNN are significantly better than Avg, GE_spec and Last. For GE_gen the available data does not allow us to assign it to one of the two groups

and KNN outperform both baselines. We can also observe that grammatical evolution approaches are better for 30 and 60 minutes time windows. This indicates that with the grammar we successfully reduced the search space, but failing for higher windows. A future solution will include a multi-objective grammatical evolution view, as suggested in previous work [16].

Execution times depends on the on the particular configuration and methods. For instance, GE_spec takes 2.4 seconds on average for obtaining a model on 100 data points using 100 individuals and 100 generations. All the configurations of GP and GE need similar times which means around 22 minutes for a model on 11500 data points with 200 individuals and 250 generations.[6] RF and KNN are faster. However, the key element in this study is the evaluation time of the models, which is extremely fast in all the algorithms that we have analyzed. In order to make a prediction we only need to evaluate a mathematical expression.

Conclusion

The control of glucose levels in diabetic patients is a hard task that is typically done with non-automated procedures. In this paper, we have presented and compared a set of evolutionary methods that are used to perform data-based predictions of glucose levels taking into account the historical data of glucose, carbohydrate intakes, and insulin injections. These methods are based on genetic programming (GP) and grammatical evolution (GE); we discuss the approaches used in this work as well as the preprocessing and feature calculation steps designed for this research. These evolutionary algorithms have been compared to two state-of-the-art machine learning methods, namely random forests (RF) and k-nearest neighbor (KNN) regression, and also two baseline predictors (Last and Avg) on a database comprising data of ten patients from the Spanish Public Healthcare System. Four prediction horizons have been studied: $t + 30$ minutes, $t + 60$ minutes, $t + 90$ minutes and $t + 120$ minutes.

Our results show that RF and KNN perform well in the forecasting of blood glucose concentration up to two hours ahead using only past glucose values as well as past and expected future insulin and carbohydrate inputs. 90% of the forecasts are considered correct (i.e., are in Clarke error categories A and B), still even the best models produce 5% to 10% of serious errors (category D) and approximately 0.5% of critical errors (category E). GE variants perform well for short term forecasts (30 minutes and 60 minutes). However, the advantage diminishes for longer forecasting horizons.

[6] Execution times using an Intel 7 processor with 8GB RAM running Windows 10 as operating system at 3.6 GHz.

GP variants and state-of-the-art machine learning methods (RF and KNN) perform well for longer forecasting horizons (90 minutes and 120 minutes).

Analyzing our results we see that the performance of our GE and GP methods is better that or equal to the state-of-the-art machine learning methods; we also see that those enhanced GE and GP variants, that have been especially tuned for blood glucose prediction, perform better than their generic counterparts. As they produce more compact models, GE and GP provide model expressions that are better suited to be embedded in wearable devices.

As future work we plan to improve our GE and GP algorithms by the hybridization with different methods. In addition, alternative approaches like the multi-objective optimization will be taken into account.

Acknowledgements This work was partially supported by the Spanish Government Minister of Science and Innovation under grants TIN2014-54806-R and TIN2015-65460-C2. J. I. Hidalgo also acknowledges the support of the Spanish Ministry of Education mobility grant PRX16/00216. S. M. Winkler and G. Kronberger acknowledge the support of the Austrian Research Promotion Agency (FFG) under grant #843532 (COMET Project Heuristic Optimization in Production and Logistics).

Compliance with Ethical Standards

Conflict of interests The authors declare that they have no conflict of interest.

Informed Consent Informed consent was obtained from all individual participants included in the study.

Ethical Approval All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Helsinki declaration and its later amendments or comparable ethical standards.

References

1. Adaptive, G. roup., B. S.: ABSys JECO (Java Evolutionary COmputation) library. Available at: https://github.com/ABSysGroup/jeco, 2016.
2. Adaptive and Bioinspired Systems Group: Java evolutionary computation library (JECO). https://github.com/ABSysGroup/jeco, 2017.
3. Affenzeller, M., and Wagner, S.: Offspring selection: A new self-adaptive selection scheme for genetic algorithms. In: *Adaptive and Natural Computing Algorithms*, pp. 218–221. Springer, 2005.
4. Affenzeller, M., Wagner, S., Winkler, S., and Beham, A.: *Genetic algorithms and Genetic Programming: Modern Concepts and Practical Applications*. CRC Press, 2009.
5. Altman, N.S., An introduction to kernel and nearest-neighbor nonparametric regression. *Am. Stat.* 46(3):175–185, 1992.
6. Bakhtiani, P.A., Zhao, L. M., El Youssef, J., Castle, J. R., and Ward, W.K., A review of artificial pancreas technologies with an emphasis on bi-hormonal therapy. *Diab. Obes. Metabol.* 15(12):1065–1070, 2013.
7. Biau, G., and Scornet, E.: A random forest guided tour. *TEST* 25(2):197–227. doi:10.1007/s11749-016-0481-7, 2016.
8. Breiman, L., Random forests. *Mach. Learn.* 45(1):5–32, 2001.
9. Clarke, W. L., Cox, D., Gonder-Frederick, L. A., Carter, W., and Pohl, S. L., Evaluating clinical accuracy of systems for self-monitoring of blood glucose. *Diab. Care* 10(5):622–628, 1987.
10. Cobelli, C., Dalla Man, C., Sparacino, G., Magni, L., De Nicolao, G., and Kovatchev, B.P., Diabetes: Models, signals, and control. *IEEE Rev. Biomed. Eng.* 2:54–96, 2009.
11. Cobelli, C., Man, C. D., Pedersen, M. G., Bertoldo, A., and Toffolo, G.: Advancing our understanding of the glucose system via modeling: A perspective. *IEEE Trans. Biomed. Eng.* 61(5):1577–1592. doi:10.1109/TBME.2014.2310514, 2014.
12. Cobelli, C., Renard, E., and Kovatchev, B., Artificial pancreas: Past, present, future. *Diabetes* 60(11):2672–2682, 2011.
13. Colmenar, J. M., Hidalgo, J. I., Lanchares, J., Garnica, O., Risco, J. L., Contreras, I., Sánchez, A., and Velasco, J. M.: Compilable phenotypes: Speeding-up the evaluation of glucose models in grammatical evolution. In: *European Conference on the Applications of Evolutionary Computation*, pp. 118–133. Springer International Publishing, 2016.
14. Colmenar, J. M., Winkler, S. M., Kronberger, G., Maqueda, E., Botella, M., and Hidalgo, J. I., Predicting glycemia in diabetic patients by evolutionary computation and continuous glucose monitoring. In: Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion, GECCO '16 Companion, pp. 1393–1400. New York: ACM, 2016. doi:10.1145/2908961.2931734.
15. Colmenar, J. M., Winkler, S. M., Kronberger, G., Maqueda, E., Botella, M., and Hidalgo, J. I.: Predicting glycemia in diabetic patients by evolutionary computation and continuous glucose monitoring. In: *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, pp. 1393–1400. ACM, 2016.
16. Contreras, I., Hidalgo, J.I., and Nuñez-Letamendía, L., A hybrid automated trading system based on multi-objective grammatical evolution. *J. Intell. Fuzzy Syst.* 32(3):2461–2475, 2017.
17. Contreras, I., and Vehi, J. *Mid-Term Prediction of Blood Glucose from Continuous Glucose Sensors, Meal Information and Administered Insulin*, pp. 1137–1143. Cham: Springer International Publishing, 2016.
18. De Falco, I., Della Cioppa, A., and Tarantino, E. *A Genetic Programming System for Time Series Prediction and Its Application to El Niño Forecast*, pp. 151–162. Berlin: Springer Berlin Heidelberg, 2005.
19. Dempsey, I., O'Neill, M., and Brabazon, A.: *Foundations in Grammatical Evolution for Dynamic Environments*. Vol. 194 Springer, 2009.
20. Demsar, J., Statistical comparison of classifiers over multiple data sets. *J. Mach. Learn. Res.* 7:1–30, 2006.
21. Doyle, F. J., Huyett, L. M., Lee, J. B., Zisser, H. C., and Dassau, E.: Closed-loop artificial pancreas systems: Engineering the algorithms. *Diab. Care* 37(5):1191–1197. doi:10.2337/dc13-2108, 2014.
22. Foundation, I.D.: IDF Diabetes Atlas 2014, https://www.idf.org/sites/default/files/Atlas-poster-2014_EN.pdf.
23. Gani, A., Gribok, A. V., Rajaraman, S., Ward, W.K., and Reifman, J., Predicting subcutaneous glucose concentration in humans: data-driven glucose modeling. *IEEE Trans. Biomed. Eng.* 56(2):246–254, 2009.
24. Hansen, B., and Matytsina, I., Insulin administration: Selecting the appropriate needle and individualizing the injection technique. *Expert Opin. Drug Deliv.* 8(10):1395–1406, 2011.

25. Hidalgo, J. I., Colmenar, J. M., Risco-Martín, J. L., Cuesta-Infante, A., Maqueda, E., Botella, M., and Rubio, J. A., Modeling glycemia in humans by means of grammatical evolution. *Appl. Soft. Comput.* 20:40–53, 2014. doi:10.1016/j.asoc.2013.11.006.
26. Hidalgo, J. I., Colmenar, J. M., Risco-Martin, J. L., Cuesta-Infante, A., Maqueda, E., Botella, M., and Rubio, J. A., Modeling glycemia in humans by means of grammatical evolution. *Appl. Soft Comput.* 20:40–53, 2014.
27. Hovorka, R., Kumareswaran, K., Harris, J., Allen, J. M., Elleri, D., Xing, D., Kollman, C., Nodale, M., Murphy, H. R., Dunger, D. B., and et al., Overnight closed loop insulin delivery (artificial pancreas) in adults with type 1 diabetes: crossover randomised controlled studies. *Bmj* 342:d1855, 2011.
28. Hyndman, R. J., and Athanasopoulos, G.: Forecasting: Principles and practice. Online textbook, 2013.
29. Keijzer, M.: Improving symbolic regression with interval arithmetic and linear scaling. In: *European Conference on Genetic Programming*, pp. 70–82. Springer, 2003.
30. Kommenda, M., Kronberger, G., Wagner, S., Winkler, S., and Affenzeller, M.: On the architecture and implementation of tree-based genetic programming in heuristiclab. In: *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation*, pp. 101–108. ACM, 2012.
31. Koza, J. R. *Genetic Programming*. Cambridge: The MIT Press, 1992.
32. Ljung, L., Perspectives on system identification. *Annu. Rev. Control.* 34(1):1–12, 2010.
33. Luke, S., Two fast tree-creation algorithms for genetic programming. *IEEE Trans. Evol. Comput.* 4(3):274–283, 2000.
34. Moreno-Salinas, D., Besada-Portas, E., López-Orozco, J., Chaos, D., de la Cruz, J., and Aranda, J., Symbolic regression for marine vehicles identification. *IFAC-PapersOnLine* 48(16):210–216, 2015.
35. Nemenyi, P.: Distribution-free multiple comparisons. Ph.D. thesis Princeton University, 1963.
36. O'Neill, M., and Ryan, C., Grammatical evolution. *IEEE Trans. Evol. Comput.* 5(4):349–358, 2001.
37. O'Neill, M., and Ryan, C.: Grammatical evolution by grammatical evolution: The evolution of grammar and genetic code. In: *European Conference on Genetic Programming*, pp. 138–149. Springer, 2004.
38. Oviedo, S., Vehí, J., Calm, R., and Armengol, J.: A review of personalized blood glucose prediction strategies for t1dm patients. *Int. J. Numer. Methods Biomed. Eng.* doi:10.1002/cnm.2833, 2016.
39. Parkes, J. L., Slatin, S. L., Pardo, S., and Ginsberg, B. H., A new consensus error grid to evaluate the clinical significance of inaccuracies in the measurement of blood glucose. *Diab. Care* 23(8):1143–1148, 2000.
40. Rechenberg, I.: Evolutionsstrategie. Friedrich Frommann Verlag, 1973.
41. Santini, M., and Tettamanzi, A., Genetic programming for financial time series prediction. In: Proceedings of the 4th European Conference on Genetic Programming, EuroGP '01, pp. 361–370. London: Springer-Verlag, 2001. http://dl.acm.org/citation.cfm?id=646809.704093.
42. Schwefel, H. P. *Evolutionsstrategie und numerische optimierung*. Technische Universität Berlin: Ph.D. thesis, 1975.
43. Sparacino, G., Zanderigo, F., Corazza, S., Maran, A., Facchinetti, A., and Cobelli, C., Glucose concentration can be predicted ahead in time from continuous glucose monitoring sensor time-series. *IEEE Trans. Biomed. Eng.* 54(5):931–937, 2007.
44. Velasco, J. M., Winkler, S., Hidalgo, J. I., Garnica, O., Lanchares, J., Colmenar, J. M., Maqueda, E., Botella, M., and Rubio, J. A.: Data-based identification of prediction models for glucose. In: *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pp. 1327–1334. ACM, 2015.
45. Wagner, S., and Affenzeller, M.: Sexualga: Gender-specific selection for genetic algorithms. In: *Proceedings of the 9th World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI)*. Vol. 4, pp. 76–81, 2005.
46. Weissberg-Benchell, J., Antisdel-Lomaglio, J., and Seshadri, R., Insulin pump therapy. *Diab. Care* 26(4):1079–1087, 2003.
47. Wilinska, M. E., Chassin, L. J., Schaller, H. C., Schaupp, L., Pieber, T. R., and Hovorka, R., Insulin kinetics in type-1 diabetes: Continuous and bolus delivery of rapid acting insulin. *IEEE Trans. Biomed. Eng.* 52(1):3–12, 2005.
48. Zhao, C., Dassau, E., Jovanovič, L., Zisser, H. C., Doyle, F. J. . I, and Seborg, D. E., Predicting subcutaneous glucose concentration using a latent-variable-based statistical method for type 1 diabetes mellitus. *J. Diab. Sci. Technol* 6(3):617–633, 2012.