Predicting Glycemia in Diabetic Patients By Evolutionary Computation and Continuous Glucose Monitoring

J. Manuel Colmenar[1]
josemanuel.colmenar@urjc.es

Stephan M. Winkler[2]
stephan.winkler@fh-hagenberg.at

Gabriel Kronberger[2]
gabriel.kronberger@fh-hagenberg.at

Esther Maqueda[3]
esthermaq@gmail.com

Marta Botella[4]
marta.botella@salud.madrid.org

J. Ignacio Hidalgo[5]
hidalgo@ucm.es

[1] Rey Juan Carlos University, Dpt. of Computer Science and Statistics, Móstoles, Spain

[2] University of Applied Sciences Upper Austria, HEAL, Hagenberg, Austria

[3] Hospital Virgen de la Salud, Endocrinology and Nutrition Service, Toledo, Spain

[4] Hospital U. Príncipe Asturias, Endocrinology and Nutrition Service, Alcalá de Henares, Spain

[5] Universidad Complutense de Madrid, Adaptive and Bioinspired Systems Group (ABSys), Madrid, Spain

ABSTRACT

Diabetes mellitus is a disease that affects more than three hundreds million people worldwide. Maintaining a good control of the disease is critical to avoid not only severe long-term complications but also dangerous short-term situations. Diabetics need to decide the appropriate insulin injection, thus they need to be able to estimate the level of glucose they are going to have after a meal. In this paper we use machine learning techniques for predicting glycemia in diabetic patients. The algorithms utilize data collected from real patients by a continuous glucose monitoring system, the estimated number of carbohydrates, and insulin administration for each meal. We compare (1) non-linear regression with fixed model structure, (2) identification of prognosis models by symbolic regression using genetic programming, (3) prognosis by k-nearest-neighbor time series search, and (4) identification of prediction models by grammatical evolution. We consider predictions horizons of 30, 60, 90 and 120 minutes.

Keywords

Genetic programming; grammatical evolution; diabetes; symbolic regression

1. INTRODUCTION

Diabetes Mellitus (DM) is a disease where patients suffer hyperglicemia due to a defect either in the secretion and/or in the action of the insulin. This happens because insulin allows the entrance of glucose inside the cells to be used. If that does not happen, glucose remains in the blood. DM is a worldwide health problem, according to the International Diabetes Federation (IDF, [9]); in 2014 more than 387 million people worldwide suffered from some type of DM.

In short, there exist two main types of DM: Type 1 (T1DM) and Type 2 (T2DM). T1DM is an autoimmune process that destroys the cells where insulin is produced. It may appear at any age, with a peak of incidence in infancy and adolescence. T2DM, the most frequently seen type, usually appears in mature adults. T2DM is related to the resistance to insulin of the cells and can be treated with oral drugs.

All types of DM require to control the glucose level in the blood. It is very important to maintain a good glycemic control to avoid not only short-term, but also long-term complications. On the one hand, prolonged high values of glucose in blood, so-called hyperglycemia, augment the risk of suffering from related health problems such as blindness, kidney affections or amputations [12]. On the other hand, extremely low values of glycemia are very dangerous since they can lead to the loss of conscience, coma and in the worst of the situations to death of the patient.

A T1DM patient must be able to calculate how much insulin is needed to process the carbohydrates associated to the meals without a dangerous increment in the glucose level. This is a common issue and the majority of DM patients either make approximations or use generic predictors. Diabetics must calculate the dose of insulin the need with the result of capillary blood samples before meals and estimating the amount of carbohydrates they are going to eat. There are another variables they consider such as exercise, stress, etc. Glucose level control is usually difficult and can be facilitated by the use of continuous glucose monitoring (CGM) systems and insulin pumps or personalized injections. CGM give the patients a lot of information to control glycemia; still, this huge amount of data should be analyzed carefully.

There exist many situations where it could be difficult to control glycemia, especially for young patients which are

GECCO'16 Companion, July 20-24, 2016, Denver, CO, USA

DOI: http://dx.doi.org/10.1145/2908961.2931734

in the main growth phase, patients with additional diseases that could influence in the glycemia, pregnant women, or even regular patients with high variability. In all those difficult cases it would be very useful to design a system able to find a custom and individualized model of the glycemia of the patient. With this model, the patient would be able to decide the amount of insulin to inject, taking into consideration the (number and types of) carbohydrates he/she is going to eat, the glucose level and even other personal considerations.

Recent research results show that there is a high degree of variability in the responses of different individuals to identical meals - which indicates that the relationship between meals, insulin administration, and blood glucose values is still by far not completely understood. From the point of view of informatics, one of the main problems in this context is the lack of formal models that can be used to predict the future values of blood glucose concentrations depending on current glucose values, carbohydrate intakes, and insulin injections. In the literature we can find numerous approaches tackling the problem of the management of DM with emerging technologies, a recent review can be found in [19].

There are several approximations to the problem. For instance, the work described in [16] is based on autoregressive methods and is focused on prediction of hyper- and hypoglycemias. In [7] some averaged models are presented, in [5] a review of minimal models is made and a parameter-based model for predicting blood glucose concentrations in patients with T1DM is derived. However, in all cases, as the prediction horizon increases, the models' predictive performance deteriorates. There is a need for applying more sophisticated techniques in order to capture the metabolic behavior of a patient with T1DM.

In this paper we investigate the performance of 4 different techniques for predicting glycemia in T1DM patients. We use data collected from real patients, namely insulin doses, glucose values measured by a continuous glucose monitoring system (CGMS) and the estimated number of carbohydrates eaten in each meal. We compare non-linear regression with a fixed model structure, identification of prognosis models by symbolic regression using genetic programming (GP), prognosis by k-nearest-neighbor time series search (kNN), and identification of prediction models using grammatical evolution (GE). Predictions are calculated for the next 30, 60, 90, and 120 minutes.

Experimental results show that symbolic regression and kNN time series techniques obtained good results in terms of correlation on test samples, especially in the 30-minutes horizon. We have identified models with GE for each of the main meals of a day (breakfast, lunch, and dinner) with good results in the test phase for some of the models reaching correlations above 0.9.

The rest of the paper is organized as follows: In Section 2 we describe the techniques and algorithms compared in this paper. Section 3 describes the results with all the four approximations for 3 real female patients. Finally, Section 4 summarizes the conclusions of this work.

2. MODELING TECHNIQUES

The goal of this research is to find customized mathematical models that predict future glucose levels ($gluc$) on the basis of current and past glucose levels, past and future carbohydrate intakes (ch), and past and future insulin injections (ins) for a given DM patient:

$$future_{gluc}(t) = f(gluc(t-2h\dots t), ch(t-2h\dots t+2h), ins(t-2h\dots t+2h), t) \quad (1)$$

In this section we describe the applied algorithms, especially how they treat the given data and how they identify models.

2.1 Non-linear Regression with Fixed Model Structure

Before trying more sophisticated time series prognosis methods we trained a non-linear regression model with a predefined model structure. In this model we used a Bateman function to model smooth uptake of carbohydrate and insulin. In this model we assume that the glucose level can be modeled solely as a a linear combination of smoothed carbohydrate (ch) and insulin (ins) inputs. Thus the model assumes the glucose level at time t depends solely on all intakes of ch and ins up to time t (inclusively). It should be noted that we are not trying to create a biologically accurate model but the model could still lead to relevant insights regarding the rate of insulin and carbohydrates uptake and breakdown.

$$\mathrm{gluc}_t = \theta_1 \underbrace{\sum_{i=1}^{t} \mathrm{ch}_i B_{\alpha_1,\beta_1}(t-i)}_{ch'(t)} + \theta_2 \underbrace{\sum_{i=1}^{t} \mathrm{ins}_i B_{\alpha_2,\beta_2}(t-i)}_{ins'(t)} + \theta_3$$

Since we are combining the smoothed inputs in a linear model we used a slightly modified variant of the Bateman function:

$$B_{\alpha,\beta}(d) = \frac{1}{\beta-\alpha}(\mathrm{e}^{-\alpha d} - \mathrm{e}^{-\beta d})$$

We optimized parameters α and β using covariance matrix adaptation evolution strategy (CMAES) [10]. The parameter vector θ is fit using least-squares regression for fixed α and β. The objective function for CMAES is the root of mean of squared errors (RMSE).

For a more efficient calculation of $\mathrm{ch}'(t)$ and $\mathrm{ins}'(t)$ in one pass over the dataset we split the Bateman function into two terms

$$\begin{aligned} \mathrm{ch}'(t) &= \sum_{i=1}^{t} \mathrm{ch}_i \frac{1}{\beta-\alpha}(\mathrm{e}^{(-\alpha(t-i))} - \mathrm{e}^{(-\beta(t-i))}) \\ &= \frac{1}{\beta-\alpha}(\underbrace{\sum_{i=1}^{t} \mathrm{ch}_i \mathrm{e}^{-\alpha(t-i)}}_{\mathrm{ch}'_\alpha(t)} - \underbrace{\sum_{i=1}^{t} \mathrm{ch}_i \mathrm{e}^{-\beta(t-i)}}_{\mathrm{ch}'_\beta t}) \end{aligned}$$

and use a recursive form incremental calculation of ch'_α and likewise for ch'_β, ins'_α and ins'_β

$$\begin{aligned} \mathrm{ch}'_\alpha(1) &= \mathrm{ch}_1 \\ \mathrm{ch}'_\alpha(t) &= \mathrm{ch}'_\alpha(t-1)\mathrm{e}^{-\alpha} + \mathrm{ch}_t \quad , \quad t > 1 \end{aligned}$$

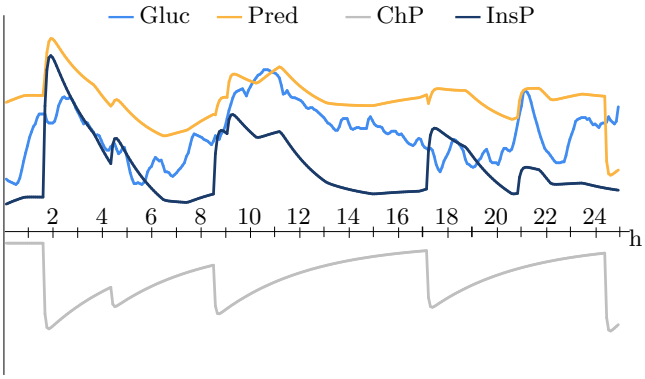

Figure 1: Fixed model outputs for one of the patients under study.

Figure 1 shows an example output for the fixed model for one of the patients under study. The smoothed functions based on insuline input and carbohydrate input are added to produce the prediction.

2.2 Identification of Prognosis Models by Symbolic Regression

Amongst other possibilities, we apply symbolic regression using genetic programming (GP) with strict offspring selection (OS) as described in [18] and [2]. The functions set described in [18] (including arithmetic as well as logical ones) was used for building composite function expressions.

Applying offspring selection has the effect that new individuals are compared to their parents; in the strict version, children are passed on to the next generation only if their quality is better than the quality of both parents. Figure 2 shows GP with OS as used in the research discussed here.

In addition to splitting the given data into training and test data, we apply GP in such a way that a part of the given training data is not used for training models and serves as validation set (VAL); in the end, when it comes to returning the eventual results, the algorithm returns those models that perform best on validation data. This approach has been chosen because it helps to cope with over-fitting; it is also applied in other GP based machine learning algorithms as for example described in [4].

We use GP as implemented in HeuristicLab [17, 11], a framework for prototyping and analyzing optimization techniques in which evolutionary algorithms as well as numerous machine learning algorithms and analysis functions are available. Figure 3 shows GP solving a regression problem in HeuristicLab 3.3.

In detail, our goal is to identify models for the following target variables that describe the future glucose values for time t:[1]

$$gluc_{f30}(t) = gluc(t + 30min) \quad (2)$$
$$gluc_{f60}(t) = gluc(t + 60min) \quad (3)$$
$$gluc_{f90}(t) = gluc(t + 90min) \quad (4)$$
$$gluc_{f120}(t) = gluc(t + 120min) \quad (5)$$

[1]The subscript "$f_{...}$" here denotes future values, where as the subscript "$h_{...}$" denotes previously recorded ("history") values.

For time t we define the following set of features $F(t)$ that describe the history of the time series (glucose, insulin, and carbohydrates) until t as well as the future intakes of insulin and carbohydrates:

$$F(t) = F_{\text{gluc}}(t) \cup F_{\text{ins\_hist}}(t) \cup F_{\text{ins\_fut}}(t) \cup F_{\text{ch\_hist}}(t) \cup F_{\text{ch\_fut}}(t) \quad (6)$$

The exact definition of the feature sets is given in the appendix.

Thus, our concrete goal here is to identify models that describe the future glucose values of time t as

$$gluc_{f30}(t) = f_{f30}(F(t)) \quad (7)$$
$$gluc_{f60}(t) = f_{f60}(F(t)) \quad (8)$$
$$gluc_{f90}(t) = f_{f90}(F(t)) \quad (9)$$
$$gluc_{f120}(t) = f_{f120}(F(t)) \quad (10)$$

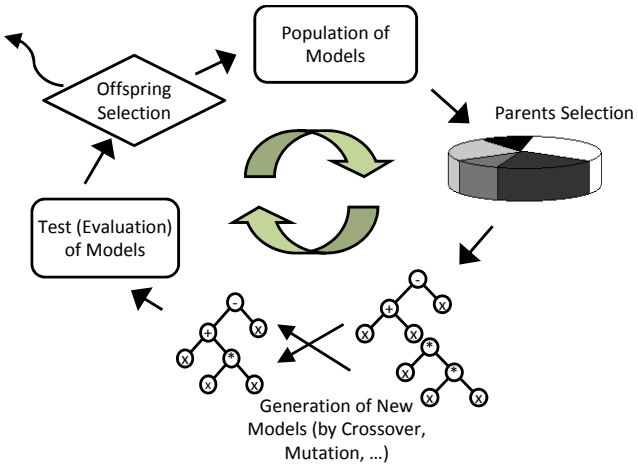

Figure 2: Genetic programming with offspring selection [18].

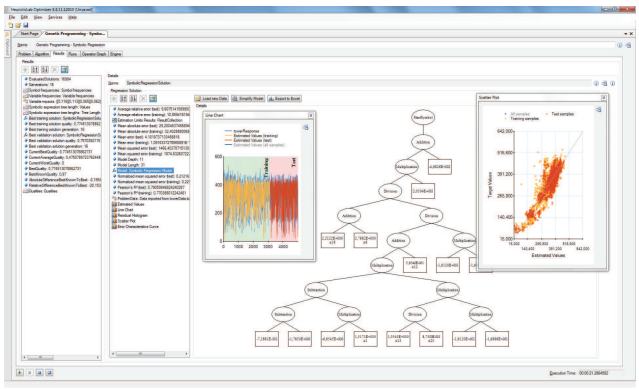

Figure 3: Solving a regression problem by symbolic regression using genetic programming in HeuristicLab 3.3.

2.3 Prognosis by k-Nearest-Neighbor Time Series Search

As an alternative we apply a model free approach for forecasting glucose values: For predicting glucose values for time stamps $t_{future} > t$ 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. I.d., we use a k-nearest neighbor approach [3] and look for time series that are most similar to the current time series (of the features described in the previous section), and predict the average of the further progresses of those similar time series as the future glucose values from t on.

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

$$X_{norm} = (X - avg(X))/std(X) \tag{11}$$

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 constant factor w_i:

$$dist(s(t_1), s(t_2), w) = dist(F(t_1), F(t_2), w) = \frac{\sum w_i * (F_{norm}(t_1)[i] - F_{norm}(t_2)[i])^2}{\sum_i w_i} \tag{12}$$

Given a set of training data time stamps TR we look for the k nearest neighbors for sample $s(t)$ as those that have a smaller distance to the target sample than the other samples:

$$nn(t, w, k, TR) : |nn(t, w, k, TR)| = k \wedge \forall_{i,j \in TR} : (i \in nn(t, w, k, TR) \wedge j \notin nn(t, w, k, TR)) \Leftrightarrow dist(t, i, w) < dist(t, j, w) \tag{13}$$

The future glucose progress from time t on is denoted as $future(t)$. The estimated future glucose progress is the average of the progresses of the similar situations, where each progress p is shifted by an offset o that is calculated as the difference of $gluc(t)$ and the first glucose value of p:

$$future(t) = [gluc(t), gluc(t + 5min), gluc(t + 10min), \ldots, gluc(t + 120min)] \tag{14}$$

$$o(t_1, t_2) = gluc(t_1) - gluc(t_2) \tag{15}$$

$$prognosis(t, w, k, TR) = \frac{1}{k} * \sum_{i=1}^{k} (future(nn(t, w, k, TR)_i) + o(t, nn(t, w, k, TR)_i)) \tag{16}$$

Figure 4 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 for $future_{gluc}(t)$.

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

$$quality(prognosis(t, w, k, TR)) = \frac{1}{n} \sum_{i=1}^{n} ((prognosis(t, w, k, TR)_i - future(t)_i)^2) \tag{17}$$

And 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

$$quality(w, TR, VAL) = \frac{1}{|VAL|} \sum_{t \in VAL} quality(prognosis(t, w, k, TR)) \tag{18}$$

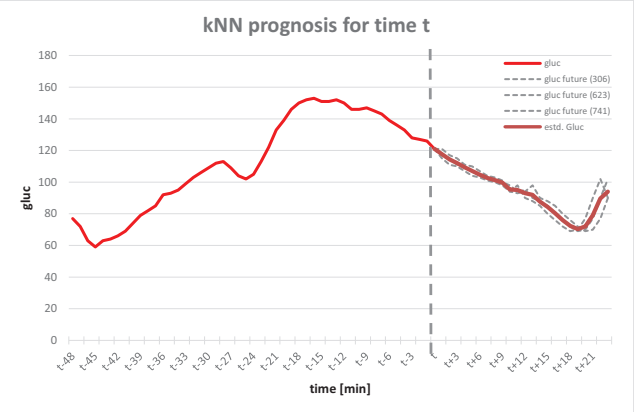

Figure 4: Exemplary calculation of $future_{gluc}(t)$ as the average of the three nearest neighbors' progresses.

For the test series documented in this paper we optimized k and w using a $(\mu + \lambda)$ evolution strategy [14, 15]:

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 1, increased by 1, or remains unchanged. The mutation strength parameter σ is in each generation adapted in dependence of the mutations' success: σ is increased (multiplied with 1.1) if more than 20% of the mutations lead to better solution quality, and σ is decreased (divided by 1.1) 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.

2.4 Grammatical Evolution

Another evolutionary approach we propose is based on grammatical evolution (GE; [13], [6]). GE is a grammar-based form of GP that combines principles from molecular biology to the representational power of formal grammars. In brief, the population is formed by a set of individuals denoted by chromosomes, whose phenotypes are obtained after a decoding process guided by the grammar. Besides, the chromosome-based representation allows the use of traditional genetic operators instead of operating on solution trees, as in standard GP.

In the case of modeling the glycemia of diabetic patients, the phenotype of an individual will be the model expression for prognosis. Therefore, we have defined a grammar to guide the optimization process toward a model expression for prediction. In short, the grammar considers that the prediction for time t may depend on the previous values of glucose, carbohydrates ingestion and insulin injection.

```
# Model expression
<func> ::= <exprgluc> + <exprch> - <exprins>

# Glucose
<exprgluc> ::= (<exprgluc> <op> <exprgluc>) | <preop>
      (<exprgluc>) | (<cte> <op> <exprgluc>)
         | predictedData(t-<idx
         >) | realData(t-<idx2hOrMore>)

# CH
<exprch> ::= (<exprch> <op> <exprch>)
         |<preop> (<exprch>)
         |(<cte> <op> <exprch>)
         |(getPrevData(1,t,1) * <cte> * <curvedCH>)

# Insulin:
## Sum of insulins in past 2h minus the peak
## Curve for the peak in past 2h
<exprins> ::= (<exprins> <op> <exprins>)
         |<preop> (<exprins>)
         |(<cte> <op> <exprins>)
         |getVariable(2,t-<idx>)

<op> ::= +|-|*|/
<preop> ::= Math.exp|Math.sin|Math.cos|Math.log
<cte> ::= <base>*Math.pow(10,<sign><exponent>)
<base> ::= 1|2|3| .. |97|98|99
<exponent> ::= 1|2|3|4|5|6|8|9
<sign> ::= +|-
<idx> ::= <dgtNoZero>|<dgtNoZero><dgt>|<dgtNoZero><
     dgt><dgt>
<dgtNoZero> ::= 1|2|3|4|5|6|7|8|9
<dgt> ::= 0|1|2|3|4|5|6|7|8|9
<ZeroOneTwo> ::= 0|1|2
```

Figure 5: An extract of the grammar developed for the extraction of models of glycemia.

The GE algorithms applied in this research are implemented in Java using the ABSys JECO library [1]. In addition, we have followed the approach of *compilable phenotypes* [8], which allows the inclusion of Java executable code as part of the phenotype in order to speed up the evaluation of individuals.

Figure 5 shows an extract of the rules that have been defined in our grammar. Next, we detail the most important ones. We construct a grammar searching for an expression based on glucose (`<exprgluc>`), plus some expression regarding carbohydrates (`<exprch>`), minus an expression of insulin (`<exprins>`). The expression of glucose denoted by `exprgluc` is a recursive rule that may produce a complex formula using arithmetic operators (`<op>`), functions (`<preop>`) and constant values (`<cte>`) which, in our case, are generated through a base and an exponent built with integer values.

The glucose expression works with a prediction window of 2 hours. Therefore, the terminal values can be either the predicted value within the window, or the real data before this window. This behavior is obtained with functions `predictedData` and `realData`, and the indexes that are formed for them: `<idxCurr2h>` and `<idx2hOrMore>`. Notice that the dataset provides data in a 5 minutes' basis, therefore, *t-24* means 2 hours ago.

Figure 5 shows some of the non-terminal symbols related to operators and auxiliary indexes such as `<op>`, `<preop>`, etc.

As can be seen, the proposed grammar defines models that work in a window of 2 hours for prediction. Hence, in the training phase we have considered slots of 4 hours where the last 2 hours of data are predicted. We have divided the data of each patient's day into six slots of 4 hours starting at 0:00 h. (12:00 am). Then, we have selected the slots that include one of the principal meals, which are breakfast, lunch and dinner, in order to obtain more accurate results.

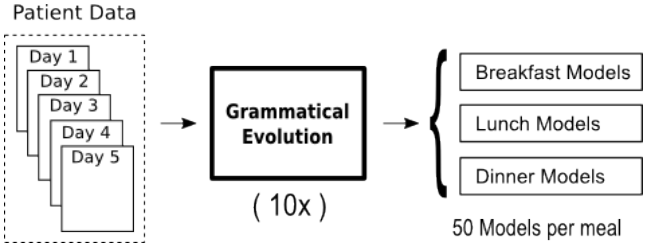

Figure 6: Training process in the GE approach for one patient.

Parameter	Value
Population	250
Generations	2500
Crossover probability	0.7
Mutation probability	0.01

Table 1: Algorithm parameters for training in GE.

The training phase consists of randomly selecting 5 days from each patient. Then, we run GE 10 times for each one of the selected slots of each day. Therefore, for each patient we have 50 models per meal, that is, a total of 150 models after training.

The value of the parameters of the genetic algorithm used in the training phase were selected after a set of preliminary experiments. Table 1 shows these values.

Figure 7 shows an example of training where a model is obtained for breakfast in one day for *patientA*. The figure shows the prediction for the last 24 data and, in this case, the model is very accurate.

In order to reduce the number of models, we have run a validation phase that follows the same idea. We have randomly selected 2 days and, for each of the slots corresponding to the three main meals, we have evaluated the models obtained in the training phase. After that, we have selected the 5 best models for each day according to the RSME value obtained. Hence, after the validation we have 10 models for each meal for each patient.

3. TEST RESULTS

3.1 Datasets

We work with the data of three real patients that we will keep anonymous. We will call them *patientA*, *patientB* and *patientC*. All the three are women, their average age is 38 years. The data were collected from a system formed by a CGMS and an insulin pump attached to the patient's body. Measures of both variables were taken each 5 minutes, and carbohydrate 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 three patients.

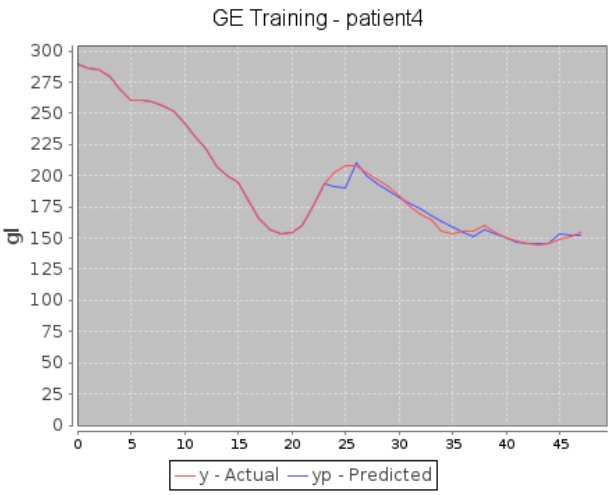

Figure 7: Evaluation of a breakfast model from training with 4 hours of data (48 values on x-axis) from *patientA* using GE. Glucose is denoted as y and prediction is denoted as yp.

Once the training phase was performed, we have tested the models that each one of the proposed techniques have generated.

3.2 Analysis

For the fixed model structure (FM), the complete dataset was divided into two halves. One half was used for training and the other half of the data was used for testing. In order to compare the algorithms, the correlation coefficient (ρ) was calculated between the original and prediction values. The ρ values were calculated for each patient separately as we tuned the model parameters for each patient separately. For the prediction of glucose at time t only the values of carbohydrates and insulin up to time t have been used, as indicated in the equation describing the model, described in Section 2.1.

In the case of GP and kNN, the correlation coefficient was calculated in the same way, but using the estimations given by the models. In GP and kNN the estimated values are obtained as the average of the models' outputs for $t+30$, $t+60$, $t+90$, and $t+120$.

Table 2 shows the correlation of original glucose values and glucose estimations calculated by models identified by FM as well as GP, and kNN, which are very good for short-term predictions. Figure 8 shows the scatter plot of predictions for *patientA* using GP, for which the best estimation (in terms of correlation) was identified.

In the case of GE, the test phase was made in a different way. We randomly selected two days of data for each patient. On each day, we have taken the time slot corresponding to a meal, and we have evaluated the models obtained in the validation for the given patient. After that, we have calculated, for each time t, the mean of the values given by the models which were within the range $(0, 400]$. This range is established by the limits of actual glucose monitors. Therefore, we consider that any prediction outside of that range is not valid.

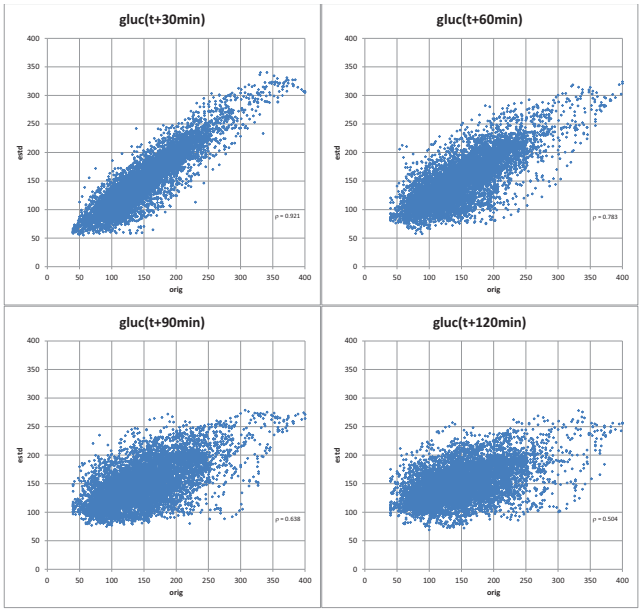

Figure 8: Overview of original vs. estimated future glucose values for *patientA*. Estimated values are calculated using models identified by GP.

		Breakfast	Lunch	Dinner
patientA	Day 1	**0.91**	-0.827	0.182
	Day 2	**0.945**	0.399	-0.415
patientB	Day 1	0.513	**0.883**	**0.97**
	Day 2	-0.172	**0.896**	-0.962
patientC	Day 1	**0.872**	-0.658	**0.961**
	Day 2	0.45	0.086	**0.949**

Table 3: Correlation (ρ) of the original values and the values estimated as the average of models obtained with GE.

The correlation values for the predictions are shown in Table 3. It can be seen that, in general, some models are not very accurate. Negative correlations mean that the relationship between the actual value and the prediction is inverse (if one grows, the other decreases). However, we obtained several values (given in bold font in the table) which are very good. In fact, the *Breakfast* models for *patientA* and the *Dinner* models for *patientC* are better than the models derived by GP and kNN.

Figure 9 shows the scatter plot of predictions for *patientB* using the GE average model for dinner on day of testing, which presents the best estimation in terms of correlation.

4. CONCLUSIONS AND FUTURE WORK

In this paper we have presented four different techniques to obtain expressions to model the glycemia of diabetic patients. We have worked on a set of data that was collected from three real patients who provided glycemia levels, insulin doses, and carbohydrate ingestions.

First, we have described a non-linear regression with fixed model structure, which obtained low values of correlation in

		$gluc_{f30}(t)$	$gluc_{f60}(t)$	$gluc_{f90}(t)$	$gluc_{f120}(t)$
patientA	GP	0.921	0.783	0.638	0.504
	kNN	0.908	0.707	0.492	0.319
	FM	0.367	0.367	0.367	0.367
patientB	GP	0.913	0.790	0.681	0.555
	kNN	0.880	0.679	0.507	0.373
	FM	0.537	0.537	0.537	0.537
patientC	GP	0.902	0.778	0.669	0.586
	kNN	0.846	0.644	0.492	0.352
	FM	0.43	0.43	0.43	0.43

Table 2: Normal correlation coefficient (ρ) between the original and the estimated values in different prediction horizons: $t+30$, $t+60$, $t+90$ and $t+120$ minutes.

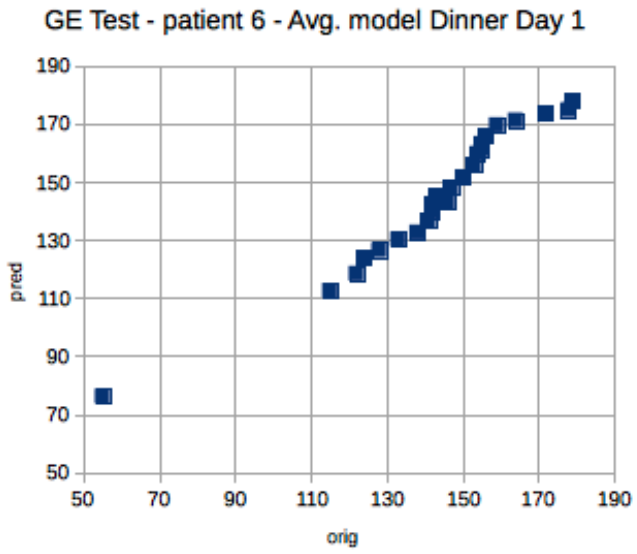

Figure 9: Overview of original vs. estimated future glucose values for *patientB* using the dinner model produced by GE on test day 1.

the test phase. However, we take this technique as a baseline to be improved. Second, we applied genetic programming based symbolic regression and k-nearest-neighbor time series search to the dataset. Both techniques obtained good results in terms of correlation on test samples, especially within a 30-minutes horizon where the correlation of original and estimated prognoses is > 0.84. Finally we applied grammatical evolution and identified models for each of the main meals of a day, namely breakfast, lunch and dinner. These results are very good in the test phase for some of the models, reaching correlations above 0.9.

As future work we will try to advance in the proposed strategies, given that this the beginning of our work with data from real patients. In addition, we will try to introduce more input variables into the patients' data such as information about the meals, the level of stress, activities, and the number of hours spent sleeping.

Acknowledgments

This work was supported by Spanish Government Grants TIN2014-54806-R and TIN2015-65460-C2, as well as the Austrian Research Promotion Agency (FFG, K-project HOPL). Stephan Winkler thanks colleagues at Bioinformatics and HEAL research groups at FH Upper Austria, Hagenberg, and Ignacio Hidalgo thanks Almudena Sánchez, Juan Lanchares and Oscar Garnica, who provided insight and expertise that assisted the research presented here.

5. REFERENCES

[1] Adaptive and Bioinspired Systems Group. ABSys JECO (Java Evolutionary COmputation) library. Available at: https://github.com/ABSysGroup/jeco, 2016.

[2] Michael Affenzeller, Stephan Winkler, Stefan Wagner, and Andreas Beham. *Genetic Algorithms and Genetic Programming - Modern Concepts and Practical Applications*. Chapman & Hall / CRC, 2009.

[3] Naomi S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[4] Wolfgang Banzhaf and Christian W.G. Lasarczyk. Genetic programming of an algorithmic chemistry. In U. O'Reilly, T. Yu, R. Riolo, and B. Worzel, editors, *Genetic Programming Theory and Practice II*, pages 175–190. Ann Arbor, 2004.

[5] Alain Bock, Grégory François, and Denis Gillet. 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.

[6] A. Brabazon, M. O'Neill, and I. Dempsey. An introduction to evolutionary computation in finance. *Computational Intelligence Magazine, IEEE*, 3(4):42–55, Nov. 2008.

[7] C. Cobelli, C. Dalla Man, G. Sparacino, L. Magni, G. De Nicolao, and B.P. Kovatchev. Diabetes: Models, signals, and control. *Biomedical Engineering, IEEE Reviews in*, 2:54 –96, 2009.

[8] J. Manuel Colmenar, J. Ignacio Hidalgo, Juan Lanchares, Oscar Garnica, Jose-L. Risco, Iván Contreras, Almudena Sánchez, and J. Manuel Velasco. *Applications of Evolutionary Computation: 19th European Conference, EvoApplications 2016, Porto, Portugal, March 30 – April 1, 2016, Proceedings, Part*

II, chapter Compilable Phenotypes: Speeding-Up the Evaluation of Glucose Models in Grammatical Evolution, pages 118–133. Springer International Publishing, Cham, 2016.

[9] International Diabetes Foundation. IDF Diabetes Atlas 2014, https://www.idf.org/sites/default/files/Atlas-poster-2014_EN.pdf.

[10] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, June 2001.

[11] Michael Kommenda, Gabriel Kronberger, Stefan Wagner, Stephan Winkler, and Michael Affenzeller. 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*, GECCO '12, pages 101–108, New York, NY, USA, 2012. ACM.

[12] Roz D Lasker. The diabetes control and complications trial–implications for policy and practice. *New England Journal of Medicine*, 329(14):1035–1036, 1993.

[13] Michael O'Neill and Conor Ryan. Grammatical evolution. *IEEE Trans. Evolutionary Computation*, 5(4):349–358, 2001.

[14] Ingo Rechenberg. *Evolutionsstrategie*. Friedrich Frommann Verlag, 1973.

[15] Hans-Paul Schwefel. *Evolutionsstrategie und numerische Optimierung*. PhD thesis, Technische Universität Berlin, 1975.

[16] G. Sparacino, F. Zanderigo, S. Corazza, A. Maran, A. Facchinetti, and C. Cobelli. Glucose concentration can be predicted ahead in time from continuous glucose monitoring sensor time-series. *Biomedical Engineering, IEEE Transactions on*, 54(5):931 –937, may 2007.

[17] Stefan Wagner, Gabriel Kronberger, Andreas Beham, Michael Kommenda, Andreas Scheibenpflug, Erik Pitzer, Stefan Vonolfen, Monika Kofler, Stephan M. Winkler, Viktoria Dorfer, and Michael Affenzeller. Architecture and design of the heuristiclab optimization environment. *Advanced Methods and Applications in Computational Intelligence, Topics in Intelligent Engineering and Informatics*, 6:197–261, 2013.

[18] Stephan M. Winkler. *Evolutionary System Identification: Modern Concepts and Practical Applications*. Schriften der Johannes Kepler Universität Linz. Universitätsverlag Rudolf Trauner, 2009.

[19] Konstantia Zarkogianni, Eleni Litsa, Konstantinos Mitsis, Po-Yen Wu, Chanchala D Kaddi, Chih-Wen Cheng, May D Wang, and Konstantina S Nikita. A review of emerging technologies for the management of diabetes mellitus. *Biomedical Engineering, IEEE Transactions on*, 62(12):2735–2749, 2015.

Appendix

Definition of feature sets:

$$
\begin{aligned}
F_{\text{gluc}}(t) = \{ & \\
& gluc_{h15}(t) = avg(gluc(t-10min), \ldots, gluc(t-20min)), \\
& gluc_{h30}(t) = avg(gluc(t-25min), \ldots, gluc(t-35min)), \\
& gluc_{h45}(t) = avg(gluc(t-40min), \ldots, gluc(t-50min)), \\
& gluc_{h60}(t) = avg(gluc(t-55min), \ldots, gluc(t-65min)), \\
& gluc_{h2h}(t) = avg(gluc(t-70min), \ldots, gluc(t-120min)), \\
& gluc_{h3h}(t) = avg(gluc(t-125min), \ldots, gluc(t-180min)), \\
\}
\end{aligned}
\tag{19}
$$

$$
\begin{aligned}
F_{\text{ins\_hist}}(t) = \{ & \\
& ins_{h15}(t) = avg(ins(t-10min), \ldots, ins(t-20min)), \\
& ins_{h30}(t) = avg(ins(t-25min), \ldots, ins(t-35min)), \\
& ins_{h45}(t) = avg(ins(t-40min), \ldots, ins(t-50min)), \\
& ins_{h60}(t) = avg(ins(t-55min), \ldots, ins(t-65min)), \\
& ins_{h2h}(t) = avg(ins(t-70min), \ldots, ins(t-120min)), \\
& ins_{h3h}(t) = avg(ins(t-125min), \ldots, ins(t-180min)), \\
\}
\end{aligned}
\tag{20}
$$

$$
\begin{aligned}
F_{\text{ch\_hist}}(t) = \{ & \\
& ch_{h15}(t) = avg(ch(t-10min), \ldots, ch(t-20min)), \\
& ch_{h30}(t) = avg(ch(t-25min), \ldots, ch(t-35min)), \\
& ch_{h45}(t) = avg(ch(t-40min), \ldots, ch(t-50min)), \\
& ch_{h60}(t) = avg(ch(t-55min), \ldots, ch(t-65min)), \\
& ch_{h2h}(t) = avg(ch(t-70min), \ldots, ch(t-120min)), \\
& ch_{h3h}(t) = avg(ch(t-125min), \ldots, ch(t-180min)), \\
& ch(t), ins(t), gluc(t), \\
\}
\end{aligned}
\tag{21}
$$

$$
\begin{aligned}
F_{\text{ins\_fut}}(t) = \{ & \\
& ins_{f15}(t) = avg(ins(t+5min), \ldots, ins(t+15min)), \\
& ins_{f30}(t) = avg(ins(t+20min), \ldots, ins(t+30min)), \\
& ins_{f45}(t) = avg(ins(t+35min), \ldots, ins(t+45min)), \\
& ins_{f60}(t) = avg(ins(t+50min), \ldots, ins(t+60min)), \\
& ins_{f75}(t) = avg(ins(t+65min), \ldots, ins(t+75min)), \\
& ins_{f90}(t) = avg(ins(t+80min), \ldots, ins(t+90min)), \\
& ins_{f105}(t) = avg(ins(t+95min), \ldots, ins(t+105min)), \\
& ins_{f120}(t) = avg(ins(t+110min), \ldots, ins(t+120min)), \\
\}
\end{aligned}
\tag{22}
$$

$$
\begin{aligned}
F_{\text{ch\_fut}}(t) = \{ & \\
& ch_{f15}(t) = avg(ch(t+5min), \ldots, ch(t+15min)), \\
& ch_{f30}(t) = avg(ch(t+20min), \ldots, ch(t+30min)), \\
& ch_{f45}(t) = avg(ch(t+35min), \ldots, ch(t+45min)), \\
& ch_{f60}(t) = avg(ch(t+50min), \ldots, ch(t+60min)), \\
& ch_{f75}(t) = avg(ch(t+65min), \ldots, ch(t+75min)), \\
& ch_{f90}(t) = avg(ch(t+80min), \ldots, ch(t+90min)), \\
& ch_{f105}(t) = avg(ch(t+95min), \ldots, ch(t+105min)), \\
& ch_{f120}(t) = avg(ch(t+110min), \ldots, ch(t+120min))\}
\end{aligned}
\tag{23}
$$