# CSE 535: Project (Group 51)

# CGM Prediction

**Janam Vaidya**
ASU, Tempe
1222377325
jvaidya4@asu.edu

**Manan Shah**
ASU, Tempe
1222696826
mshah64@asu.edu

**Harsh Patel**
ASU, Tempe
1222872391
hpatel66@asu.edu

**Dalton Turner**
ASU, Tempe
1212546372
dcturne4@asu.edu

## Abstract

Continuous Glucose Monitoring (CGM) sensors allow us to monitor glucose levels in humans in the gap of every 5 minutes. The data so obtained can be of a huge help when it comes to managing Type 1 and Type 2 Diabetes. There is usually a rise in glucose levels after a meal and the body reacts by releasing insulin to bring it back under control. This process can be used to our benefit in managing type1 diabetes as the people with type1 diabetes can't release insulin and thus insulin levels do not stay in control. In this project, we predict Meal or No Meal using the data given to us. Algorithms like SARIMA, LSTM, RNN have been used. We have used LSTM and GRU based approaches for RNN. For meal prediction, a 2 step process was followed, which included SARIMA and LSTM which gave us the timestamp at which the meal will occur.

**Keywords**

Continuous Glucose Monitoring, RNN, LSTM, SARIMA, Meal Prediction, Kalman Filter, GRU

## Introduction

There is a risk of patients that have type1 diabetes to get Hyperglycemia, a condition in which insulin secreted is not enough to convert the high level of glucose to energy and such a high level of glucose can be dangerous. There are efforts going on so that the bodies of the people with this condition can maintain glucose levels. Meal detection and prediction is a very important aspect that is used to make systems capable of releasing insulin in the body when needed for which CGM sensors are used so that the data obtained can help us make such a system. The current methods that adjust the infusion of insulin based on the patient's meal taking habits have a drawback as the risk of inconsistent information or falsely reported data persists. To overcome these drawbacks, we can develop an autonomous CGM meal detection and prediction system that handles the infusion on its own. Our project has a similar goal where the raw data collected from the sensors (CGM and bolus) is used without any interventions on machine learning models to analyze and make it capable of meal detection and prediction. If the system becomes capable enough to detect the meal on its own, a decision on insulin infusion can then be taken.

## Data Pre-Processing

The data is collected from 2 sensors, CGM sensors and bolus sensors and we explain how we have pre-processed this raw, original data. Columns of data and their significance:

**actBolusDelivered:** Amount of insulin bolus injected.
**dateMuBolus:** Timestamp for when the bolus got delivered.
**dateNumber:** Timestamp for when the glucose values were monitored. Glucose values were measured every 5 minutes as each timestamp was at a difference of 5 minutes.
**numCGM:** Glucose value monitored at a particular timestamp.

Approx 55,000 samples and 44000 samples were there in the CGM time series data and bolus data respectively. The timestamp for data from both sensors was not synchronized since they work independent of each other. We fixed this by finding the nearest bolus data timestamp from a particular CGM data timestamp and synchronising our data. There was a minor difference in the timestamp, a few seconds or milliseconds but that won't affect much of our results and hence we ignored that. Further we removed inconsistent data and NaN data was dropped as well. The final data had around 35,000 samples with 4 columns (DateCGM , ValueCGM, DateBolus, ValueBolus), which corresponded to a single timestamp and were synchronized.

## Methodology

### SARIMA

Seasonal Autoregressive Integrated Moving average (SARIMA) [1] is a model with an extended support to capture seasonal components for the time-series data especially useful when it comes to non-stationary series as the date's trend and seasonality can be captured despite the fact that it does not fluctuate around some same mean or variance. We used SARIMA to create a model that could predict CGM values for a given time series and the actual CGM values and the predicted values were compared using a threshold and a meal, no meal instance was classified for a particular timestamp.

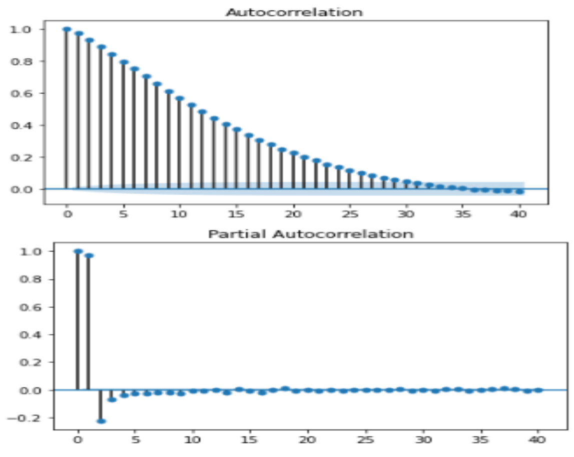

Figure 1. Autocorrelation and Partial Autocorrelation plot

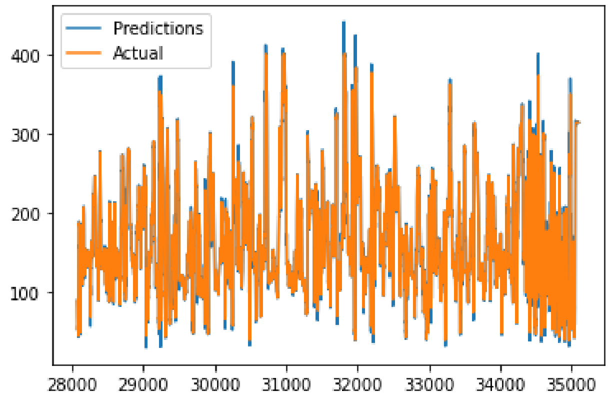

Figure 2. Actual vs. Predicted CGM values for whole test data

SARIMA works on hyper-parameters; for trend elements-> p (trend auto-regression order), d (trend difference order), and q (trend moving average order); for seasonal elements -> P (seasonal auto-regressive order), D (seasonal difference order), Q (Seasonal moving average order) and m (number of time steps for a seasonal period) which we have to decide for better results and for deciding these hyperparameter values, we plotted Auto-correlation and Partial auto-correlation as seen in Gig. 1. ACF gives the insight on how the current value is related with the past values and PCAF finds the lag for the next value. We later tweaked the parameters to generate a model that gave good predictions for the CGM values.

Quality of the model was checked using AIC values because AIC values provide us with an insight about the relative quality of the statistical model with the model with the lowest AIC values being considered the most optimal. We then ran the model on the test data which gave us the CGM predictions for each timestamp. We see in Fig. 2 that the predicted values and the actual values are very close. We also plot only 1 hour of data to check the performance. Fig. 3 shows more detailed predicted and actual values. We further check accuracy by finding mean absolute error (MAE), mean squared error (MSE) and root mean squared error (RMSE), values displayed in table 1, from which we see that the MSE error is pretty low which shows that the difference in actual and predicted CGM values is small. We use a threshold value to predict Meal or No Meal.

| Model | MAE | MSE | RMSE |
|---|---|---|---|
| Train | 6.54892 | 256.12495 | 16.0039 |
| Test | 3.08430 | 28.47469 | 4.94719 |

**Table 1. Results for SARIMA Model**

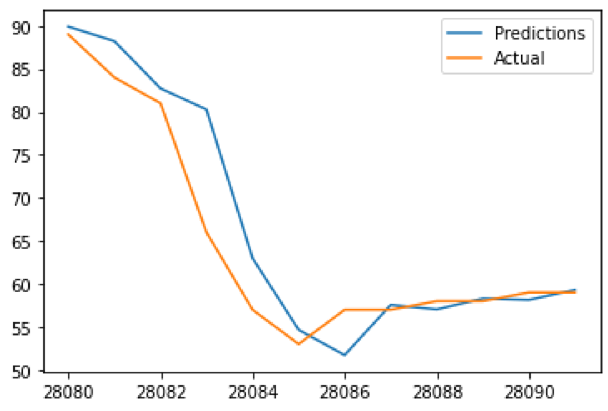

**Figure 3. Actual vs. Predicted CGM values for one hour of test data**

## KALMAN FILTER

Kalman filter is used to estimate the unknown variables using an algorithm which takes into account the measurements observed over time as well as their corresponding noise and inaccuracies. It helps in estimating the values better than standalone measuring techniques as it does the estimation considering the estimation of the unknown variables using the measurements which are obtained from the continuous monitoring and analyzing the data obtained. The algorithm used for the estimation is the Kalman filter approach and it considers the noise and the errors present in the data and the prediction of the same. The precision and the performance of this model increases significantly as the estimation of the data variables to be predicted is done keeping in account the output values of the previous time instance which helps in better prediction with joint probability distribution on the data rather than just a single instance being analyzed and the predictions being computed on the basis of the input variables of the particular data point. The process of the Kalman filter usage is done with the help of Unscented Kalman filter which helps in linear and non-linear estimations on the basis of the input variables fed to the algorithm for the prediction. The filter is primarily used for the classification for a ‘Meal’ or ‘No Meal’ for the set of input values. Below, are mentioned the algorithm in the form of equations which are used for prediction of the Continuous Glucose Monitoring values.

For a particular time instance x(k), we use the values of the previous states to estimate the value of the future state. The value depends not only on the variables of the input but also on the output of the previous time stamp. The unscented filter is used for the input values of p1, p2, p3, p4, V, and T(tau) that are used in the model. And varying with time for tuning the model which is extracted for the proposed equation in [2].

$$G_s(K+1) = h\left[p_1(k)\left(G_b(k) + \frac{G_s(k)}{hp_1(k)}\right.\right. \quad (1)$$

$$\left.\left. | -G_s(k)\right) - (k) + R_a(k)\right]$$

Using the above equation the nonlinear space model for unscented kalman filter is described as below

$$x(k+1) = f(x(k)) + w(k)$$
$$y(k) = g(x(k)) + v(k)$$

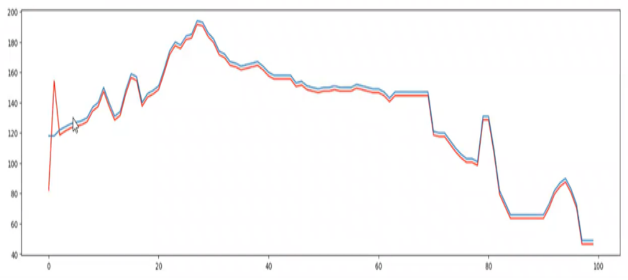

**Figure 4. Actual (Red) vs Predicted (Blue) Glucose values from Kalman Filter implementation**

With the model, we achieve an accuracy of 94.166% for the predictions. Fig. 4 depicts the values in a visual form for better analysis of the same.

### Recurrent Neural Network

For the predictions of the meal and no-meal, we have used the recurrent neural networks. We have implemented 2 variants of the RNNs for the analysis and prediction. The prediction and analysis of the data is initiated with the pre-processing of the dataset. The predictive performance of the RNNs is compared and the best predictive model is chosen for the further analysis and the working on the dataset.Below are the pre-processing techniques used for the particular form of data.

**Temporal Difference**: We transformed the data into a series of differences between sequential observations in a difference order of 1 to stabilize the time series and eliminate the temporal dependence on scale.

**Normalization:** This step reduces the sparsity present in the data for the neural networks as they do not perform with sparse forms of data. The normalization of the data is done to the scale of [-1,1] for better assessment and coverage of the maximum set of values that can be analyzed.

**LookBackTimeSteps:** The series of time sequences which are processed for temporal agreement on the data points and their respective processing are separated into minute portions with a look back for the prediction of the values incorporating the previous values of the prediction. The number of each of the look backs and the prediction for a certain amount of values is a part of the parameters which can be tuned for better results.

After the data pre-processing for RNN, specifically for GRU we have introduced a static layer in the model which takes in static features of the data as input.The static features provide extra information which the model learns in order to help in predicting the consecutive values by analyzing the trends in the static features. We have derived several static features from the data and each of the features is explained below:

**Mean:** It shows an overall measure of blood sugar level over a certain period of time, in our case over the input sequence. It depicts the value measure of the blood sugar level for a course of interval as mentioned in the continuous glucose monitoring systems.

**Median:** It represent the middle glucose value that divides  the two half and gives and idea of the probability distribution of glucose values. This value represents the middle glucose values along with the probability distribution of the prediction that has to be made based on the glucose values observed.

**Max and Min:** The min and max values report the minimum and maximum values of glucose over a time interval along with the information stating the instance with the maximum difference due to the change in the variables which are being analyzed. The extent of change in the values can be analyzed for detailed analysis of the time stamp.

**Standard Deviation:** This value is calculated to understand the deviation of the amount of variation of the glucose values. This value is considered important as the value gives an idea of the deviation and the stability of the values over a given interval of time.

**Coefficient of Variance:** It normalizes the standard deviation with the mean which indicates the spread of the input time sequence. The coefficient of variance is used for the normalization of the standard deviation noted in the sugar values over a period of time which in turn indicates the spread of the time stamps over a given sequence.

**Mean Amplitude of Glycemic Excursion (MAGE):** This parameter can be very important as it measures glycemic variability. MAGE calculates local minimum and maximum value of CGM and assists them against standard deviation.

**Glucose Fluctuation Index (GFI)**: It is the ratio of mean glucose values to the glucose coefficient of fluctuation which measures the glucose fluctuation in blood.

The RNNs face the problem of short term memory [3] which means that the sequences don't remember the information over a long sequence of layers in the neural network which results in lesser performance level than expected. For solving the problem of vanishing gradient, we use the Gated Recurrent Unit [4] and Long Short Term Memory [5] which are more complex and help in remembering the information over a long sequence of layers and are comparatively faster and efficient in the computation process. The major difference between the GRU and the LSTM is the coupling of the gates which are present which in turn requires lesser memory for the computation and has comparatively faster computation efficiency. The advantage of the LSTM over the GRU is that the LSTM outperforms with longer sequences whereas the GRU does not transfer the gradient information precisely.

***Using Gated Recurrent Unit (GRU)***

We execute the neural network architecture with multiple GRU layers which have static features extracted from the data. The extraction of the features is not unlike the normal machine learning and deep learning models mentioned so we directly have the time shifted data passed into the GRU and the learning and analyzing of the inputs is done directly by the RNN with the help of learners present in the layers of the network. The illustration of the model states the 2 layer vertically stacked GRU for the present architecture. This is mainly used for the encoding of the time series data which has to be processed as a part of the time sequence for the look backs and learning from the previous states. External feeding of the features is helpful which includes the time series data points as it would help boost the learning and increase the performance level of the network architecture. Train test split is applied with 80-20 ratio for better training of the model on the time series data and learning along with the tuning based on the gradient obtained from the analysis. The training of the model was executed with 50 epochs along with the batch training size to be 48. The moderate size of the batch makes it efficient for the learning and tuning of the model based on the parameters. We also adjusted the other hyper-parameters which includes hidden units of GURU which was kept to 120, adam optimizer along with the various dropout values which help in identifying the best values of the dropout. The optimizer is used for optimizing the best learning rate possible for the neural network model with the help of training continuously and learning from the previous results. After the training was completed we got the loss value as 0.0132,the r-squared error was observed as 0.3106 and the mean absolute value turned out to be 0.0223. Fig.5 represents the results from a particular time sequence of 4 hours from the testing set, it can be observed that as the time moves forward the prediction tends to be less accurate, this can be a problem when predicting for longer sequences. As the sequences will get longer with various instances being fed to the model, this will help prediction of the meal with the same technique of threshold used previously.

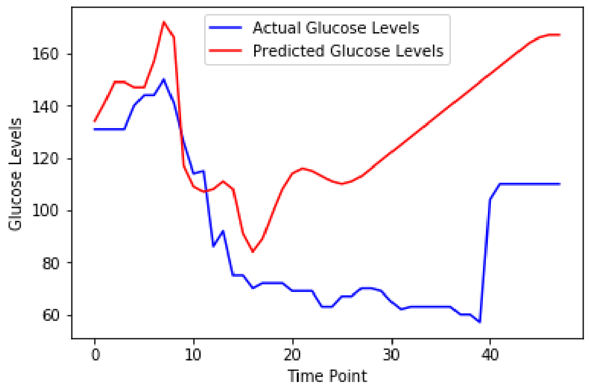

Figure 5. GRU model predictions in a span of 4 hours

### *Using Long Short Term Memory (LSTM)*

The best performing RNN is the LSTM which is executed without the use of the static features layer along with the stacked 3 layers of LSTM with the dense layer as well. The LSTM does not have gates explicitly as mentioned and included in the RNNs. The time shifting of data is not done instead the prediction of future value which is the added to the present input sequence for the estimation and prediction of the next future value of the time sequence. The splitting of the dataset into the train and test is done with the ratio .8:.2 and the data is grouped along the time and date which helps in better analysis. The dataset consists of multiple data points which portray the continuous values of the CGM for an entire day. Training of the architecture is done with batch size of 32 which is a moderate size for the training along with the total epochs of 100. The loss value which is calculated for the training set of the data is 0.0146%. The threshold values for the meal are decided and the accuracy was 62.5% for the same using the LSTM model for the predictive modeling and analysis related to it.

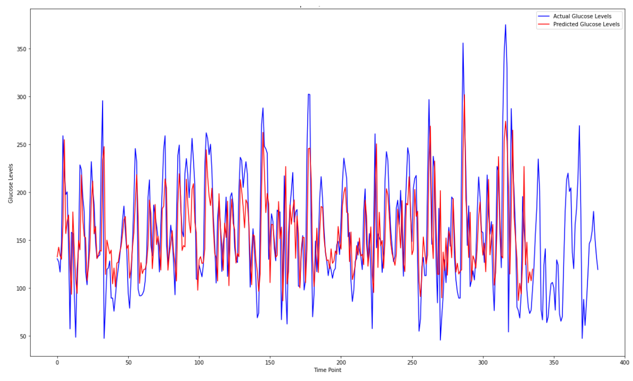

Figure 6. Actual vs Predicted Glucose values from LSTM implementation

## Meal Prediction

This section is dedicated to the explanation of the model architecture for the input data with time t and the model predicts if the next meal for the particular set of values will be after t+20, t+25 or whatsoever may be the value for the same. For the situation when the meal not being predicted in a particular future time span, the model must predict and confirm that the next meal will not occur till the next 50 time stamps.

For developing the meal prediction model, we adopted a two step process which is shown in Fig.7, firstly when the input until time t arrives it is passed to the SARIMA or LSTM detect meal or no meal instances at each time step.upto lets say time t + 100, after that the predicted glucose prediction model which predicts the future glucose values values are passed to another

LSTM model which is trained to For the development of the model for the particular use case, the use of a 2 step process is adopted firstly with the input for the time sequence t is inputted to the LSTM model.

The input to the prediction model is a sample which is grouped by date and hour, we also analyzed the trend of consecutive sequence of collected glucose values for the whole data. The data is inputted and grouped with the time stamp, and analyzed for the consecutive sequence of the data present.We kept the maximum prediction sequence equal to 33 which represents continuous data of 2.75 hours. Each value in the input data has a corresponding label Meal or No Meal attached to the glucose value. The data was divided into train and test with a ratio 0.8 which is similar to the input data of other LSTM models that we implemented. The data splitting is done into the training and test data with the decided ratio. The embeddings of the corresponding data and the network and the size of the embedding is 20. In this model, we also add the Conditional Random Fields (CRF) layer at the end after the dense layer which predicts the meal and no meal label [6]. The batch size was kept to be 128 and the model was trained for 3 epochs. It can be observed from the graph that the model is slightly overfitting but this is because there are a lot of no meals instances as compared to the meal instances and this makes it difficult for the model to detect the meal instances.

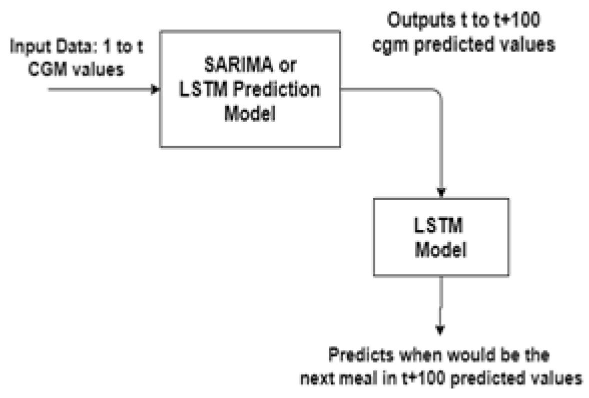
Input Data: 1 to t CGM values
SARIMA or LSTM Prediction Model
Outputs t to t+100 cgm predicted values
LSTM Model
Predicts when would be the next meal in t+100 predicted values

Figure 7. Two step prediction process

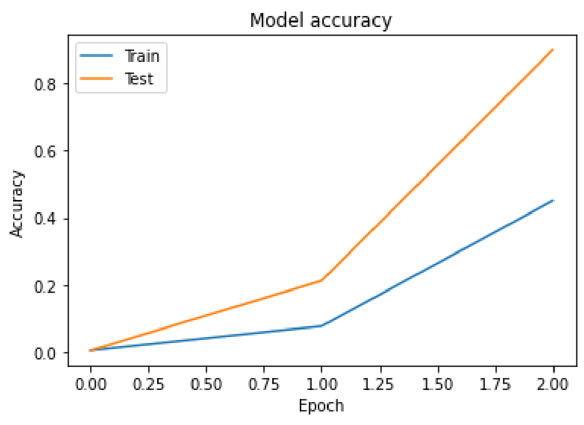

Figure 8. LSTM Prediction Accuracy Plot

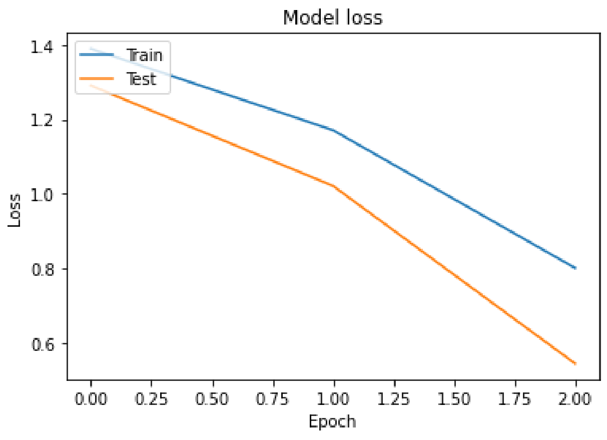

Figure 9. LSTM Prediction Loss Value Plot

## CONCLUSION AND FUTURE WORK

The project implemented works on the meal prediction based on the CGM values noted and analyzed continuously. The transformers are used for encoding the data we have which the glucose sequence and uses the self-attention mechanism for learning the complex patterns of the data along with better prediction results. The use of threshold technique is implemented for the meals and the results to be more robust to the inconsistencies in the data along with exploring of multiple other threshold techniques which can be utilized for the same. With the use of the LSTM and the GRU in the RNNS, we are able to solve the problem of missing gradients which is prevailing when dealing with long sequences of data while training a neural network architecture. The performance metrics show that the model analyzes and predicts the meals to the best observed extent to date. The future work consists of a better performing neural network for the prediction along with a better threshold techniques which helps in the precise value for the same.

## REFERENCES

[1] R. J. Hyndman and G. Athanasopoulos, “8.9 seasonal arima models,” Forecasting: principles and practice.oTexts. Retrieved, vol. 19, 2015.

[2] K. Turksoy, S. Samadi, J. Feng, E. Littlejohn, L. Quinn, and A. Cinar, “Meal detection in patients with type 1 diabetes: a new module for the multivariable adaptive artificial pancreas control system,” IEEE journal of biomedical and health informatics, vol. 20, no. 1, pp. 47–54, 2015.

[3] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” in International conference on machine learning, 2013, pp. 1310–1318.

[4] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” arXiv preprint arXiv:1409.1259, 2014.

[5] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with lstm,” 1999.

[6] Z. Huang, W. Xu, and K. Yu, “Bidirectional lstm-crf models for sequence tagging,” arXiv preprint arXiv:1508.01991, 2015.