Allan Rivalles Souza Feitosa

allanrivalles@hotmail.com

The Challenge:

- Create a computational model to forecast the 60 minutes ahead glucose of a person; given glucose, activity and heart rate past data.

Pre-processing:

During the visualization of glucose data, I noticed some missing data. The decision was to ignore any instances that "touch" those gaps. This decision was taken for a faster way of doing the experiments, but it is possible to test some approaches to fulfill those gaps like the same predictive algorithms or a linear combination of the known values.

Besides, the time stamps and the intervals among them for both activity and heard data did not match the glucose ones. Therefore, the decision was to aggregate them since the last glucose known value until the current and, thus generate aggregated activity and heart rate points matching the glucose measures.

The sliding process takes 'window_size' values from the time series as input for the model and the immediate subsequent 'horizon' values as desired output for the model. This 'window' is slid through all the time series resulting in two matrices containing the X and Y values, respectively. In order to find the best 'window_size', there were tested values from 48 to 148 with steps of size 4.

Two weeks of data were taken to train the models and 5 weeks to test. In the necessary cases (MLP and LSTM), it was used 10% of the training data as evaluation set.

Testing techniques

The decision of testing simpler techniques together with a deep learning approach was that if a simpler technique, like Linear Regression solves the problem it is computationally less expensive than a deep learning, especially to training, and may reduce the costs with cloud processing for example.

The tested techniques were Linear Regression, Multi-Layer Perceptron, Regressive Decision Tree and Long-short Term Memory. For a matter of time of processing and computational limitations, I only tested the default hyperparameter configuration of each one. Of course, it is necessary to exhaustively approach a set of hyperparameters configurations (grid search) and execute each configuration more than once to be suitable for comparation by statistical hypothesis tests.

Results

The results showed Linear Regression as the minor RMSE and Bigger Clarke A + Clarke B. However, in an eventual hyperparameter exploration of the other techniques this scenario may change.

In order to visualize the results, as there is not only one prediction for one instance (horizon = 12), two approaches were created for plotting the predicted versus true data and to calculate the Clarke for the 5 weeks of test prediction.

The approach for plotting was built by lining up predictions. First, one whole prediction (12 points) is taken and then, the prediction in which the horizon starts immediately after the 12 points is put in sequence and so on, until the end of the predictions.

The approach to calculate the Clarke error was similar, but instead of generating a new time series, the Clarke error metrics (A,B,C,D and E) was calculated for each of the predictions lined up.

Instructions about how to run the code

1 - Install python 3.7 and the requirements contained in 'requirements.txt'

2 - With the dataset '.csv' files in the same directory, run the script '1 – Data exploration…'. It will generate the graphics of the features and the data frames in '.pkl' files.

3 – Run the '2 - Testing Forecasting Techniques…'. It will read the '.pkl' files, test regression techniques and write a 'results.csv' containing the results of those tests.

4 – Run the '3 - Testing DeepLearning…'. It will test the LSTM techniques and write a 'results_deep_learning.csv' file containing the results for deep learning.

Third Party's resources used:

- Clarke Error calculation, by Trevor Tsue, under MIT License. Available in: https://github.com/suetAndTie/ClarkeErrorGrid