# BLOOD GLUCOSE FORECASTING BASED ON CGM DATA

*Gustaw Żyngiel (s192673), Krystof Spiller (s200360), Veniamin Tzingizis (s192851)*
*in cooperation with Hedia - Mads Jakobsen*

DTU Compute, Technical University of Denmark

## ABSTRACT

Forecasting of blood glucose (BG) is important in order to prevent hypo- and hyperglycemia and associated problems for people suffering from type 1 diabetes (T1D). Continuous glucose monitoring (CGM) allows people to monitor their BG levels (BGL) in usually five minute intervals quite accurately. Predictions are made 30 minutes into the future from CGM measurements, exogenous insulin and carbohydrate (CHO) intake from the in vivo Ohio T1D dataset with the deep learning framework from Hedia which preprocesses the data, trains the model, optimizes hyperparameters and evaluates the results. The model that achieved the best results was a convolutional neural network (CNN) with dilations. It is evaluated mainly in terms of RMSE and compared to the results from Hedia evaluated in the same way and to other models from the literature where the evaluation methodology is not fully transparent and a direct comparison of the results is therefore hard to perform. Our best model achieved RMSE of 20.401 compared to the best Hedia model that achieved 20.027 and the GluNet model from literature that achieved 19.28.

***Index Terms***— Neural networks, diabetes, dilated convolutions, blood glucose forecasting, continuous glucose monitoring

## 1. INTRODUCTION

### 1.1. Problem domain

Diabetes is a chronic disease that affects an estimated 463 million people around the world as of 2019 [1]. There are two main types of diabetes: Type-1 diabetes (T1D) and Type-2 diabetes (T2D). For people with T1D, exogenous insulin administration is required to manage BGL in the target range (70-180 mg/dl) to avoid hypoglycemia ($<$ 70 mg/dl) or hyperglycemia ($>$ 180 mg/dl). CGM technology provides real-time monitoring of the current glucose so people can observe their glucose concentration trend visually on devices. A solution that can predict the glucose levels in the blood can improve the lifestyle of many diabetics or even save their lives.

One can find many deep learning-based solutions for blood glucose prediction in the literature. There are instances that use fully connected neural networks (NN), (dilated) convolutional NN (CNN) and recurrent NN (RNN). Good results have been achieved with dilated CNN, such as in the case of GluNet [2], that is personalized and evolves when new personal data becomes available.

### 1.2. The goal

The main goal was to develop a model providing accurate 30 minutes forecasts of BGL, allowing to inform about the future rises and drops leading to either hyperglycemic or hypoglycemic states. Having such accurate prediction could be used as an indicator for the proper insulin and CHO intake management of the patients.

### 1.3. Framework and approach overview

The methods and models that were explored are *fbprophet*, *fastai*, dilated CNNs and RNNs with LSTMs. Our best performing model is a quite simple dilated CNN. The model and the source code can be found in GitHub: `https://github.com/Laegas/02456_deeplearning_cgmforecast`.

Using the Hedia deep learning framework (HDLF), it was rather quickly found that the best results are achieved by optimizing the simple model rather than coming up with more complex architectures that always underperformed compared to the simple model, despite utilizing sufficient training time. The HDLF uses the Ray Tune library [3] for optimizing the hyperparameters. Hyperparameters that were tested are kernel sizes, paddings, dilations, number of hidden units, activation functions, learning rate and weight decay for L2 penalty.

The HDLF evaluates the results in terms of RMSE, MARD, MAE, Clarke and Parkes error grids, precision, recall, F1 score and lag time. These are elaborated upon in subsection 3.1.

### 1.4. Data processing and testing procedure

The input used for the model development was the measurements of 6 subjects from the Ohio T1D clinical dataset [4]. Each patient's data consisted of the blood glucose (CGM) measurements taken every 5 minutes, together with the notes on the CHO and insulin intakes. On top of that, an extra column named "CGM delta", containing the CGM difference between the consecutive measurements, has been added, which is an instance of feature engineering. The rows with the missing BGL measurements have been assigned the estimated BGL value using the first degree polynomial interpolation technique. The missing values for the CHO and insulin intakes have been set to 0.

The models have been tested in two ways:

- generalized: a static set of hyperparameters was being used for all patients
- personalized: the model's hyperparameters were tuned for each individual patient

In both model development variations, a monthly record of data has been used for training the models, the data from approximately 7 consecutive days has been used for models' validation and the predictions have been conducted for the next 3-7 days.

## 2. MODEL DEVELOPMENT

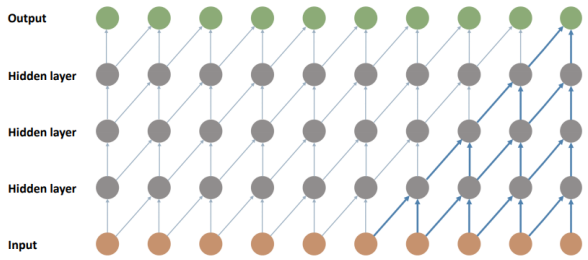
The team decided to follow two model design approaches: one focused purely on the usage of the single dimensional convolutional layers and the other one mostly focused around the LSTM usage. The obtained results were compared with the best results obtained by Hedia and also to the results from the analysed papers. On top of that, the initial idea for model verification was supposed to include the results comparison with the outputs obtained by using the 3rd party tools: *fastai* and *fbprophet* libraries.

### 2.1. Main model: CNN approach

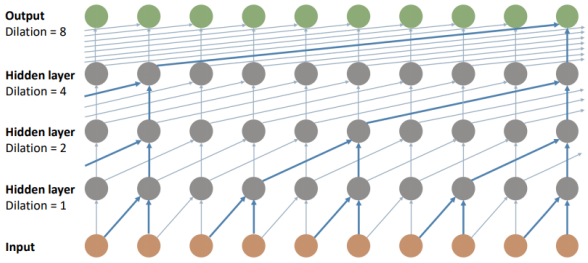
The model that generated the best results was found to be a CNN based model with the use of dilations. The most relevant hyperparameters that seem to affect the results are the number of convolution layers and the number of hidden units. Having less than 4 convolution layers made the predictions too jittery, while using more than 5 made the model too smooth. The smoother the model became, the worse it performed in predicting the highs and lows which is especially important for the glucose monitoring problem. In contrast, the more jittery the model became, the more it overshot in predicting those cases. The same behavior was observed for the size of the hidden units in the convolution layers, where the effects were more pronounced for each individual case.

After trial and error, the best architecture turned out to be 5 convolution layers with the number of hidden units ranging from 64 to 16. Specifically, 64 units for the first 3 layers, 16 for the 4th and the last layer has only one hidden unit for the prediction. The usage of techniques such as dropout, pooling and batch normalization did not improve the model's performance. Other activation functions were tried apart from ReLU: ELU [5], CELU [6] and hyperbolic tangent function, but none performed better than ReLU in terms of results and training speed. In addition, weight decay and learning rate were the hyperparameters that seemed to affect the results mostly on the individual case level, meaning for each patient.

Dilation significantly improved the model performance. Dilation provides a cheap way to increase the receptive field in a few layers without significantly increasing the number of parameters to be optimized as would be the case for increasing the receptive field with increasing the kernel sizes, see Figure 1. The best combination of dilations was found to be 1-1-2-2-4 for each convolution layer, respectively.




(a) Without dilations. A stack of standard convolutional layers with a receptive field of 5.




(b) With dilations: A stack of dilated convolutional layers with a receptive field of 16.

**Fig. 1**: Visualization of a stack of standard convolutional layers (a) and dilated convolutional layers (b): each with 3 hidden gray layers. Input marked with orange dots, output with green. Figures taken from [2].

The model in Figure 2 is the generalized model that is tuned in order to perform well for all patients. However, it was empirically observed that it performed better for

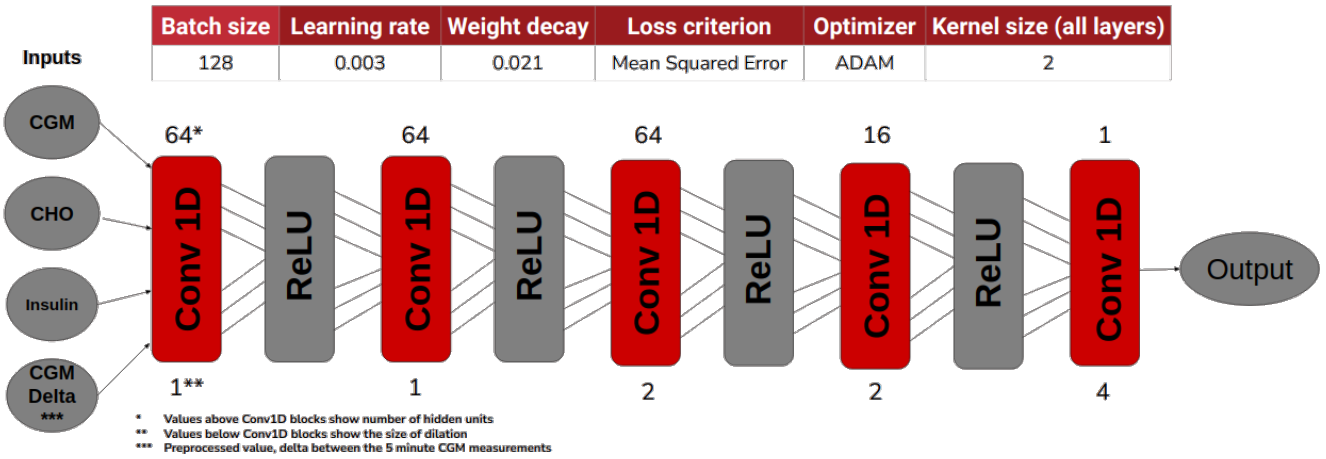



**Fig. 2**: Final CNN-based model architecture

individual patients while using certain hyperparameters. To further explore this phenomenon, two patients were taken and the model hyperparameters were changed several times. It was observed that when the results were good for one of the patients they were not so good for the other patient, for the most part. This testing method was extended to all patients and the same behaviour was seen. This was validated many times with different hyperparameters and comparing the different patient results. To sum up, tuning the model to each individual patient — personalizing it — gave the best possible results overall.

For this reason the model was tuned for each individual patient with unique hyperparameter set that performed the best for that case. The hyperparameters that were tuned are the number of hidden units of the layers, the learning rate, the weight decay and the dilations. The result is a personalized model which consistently beats the generalized model across all patients.

## 2.2. Other models and approaches

### *2.2.1. LSTM-focused architecture*

Another model architecture that has been tested was focused around the usage of the LSTM layer together with the preceding one-dimensional convolutional layers and the proceeding linear layers. The team decided to experiment with that architecture considering the common usage of RNNs in areas of time series forecasting and the fact that it has already produced decent results [7].

The model was focused around the usage of a unidirectional LSTM neural network, with either one or two recurrent layers. The input that was given to that layer was an output of a series of single-dimensional non-dilated convolutional layers, consisting of 1 to 3 layers. Its output was later processed by between 1 and 3 linear layers.

Throughout the model experiments, the team used different number of convolutional layers, changed convolutional layers' kernel sizes, dilations, paddings, activation functions and number of output channels. For the linear layers' part, the number of layers and outputs have been experimented with. Moreover, experiments with both convolutional and fully connected layers included the trials with the addition of the regularization techniques, such as dropout and batchnorming.

Unfortunately, in all the configurations tested, the LSTM-focused model has not produced the results outperforming the simpler model built around the dilated convolutional layers from subsection 2.1.

### *2.2.2. Attempt utilizing fastai library*

*Fastai* is a deep learning python package from *fast.ai* research group, co-founded by Jeremy Howard and Rachel Thomas [8]. The initial idea was to use this library to train the model and compare its results with the final model that the team would have come up with.

Unfortunately, the tool did not seem to fit our problem-related challenges directly. Both *fastai* and *fastai2*, released in July/August 2020, do not have an official library used for time series forecasting. Our team researched the *fastai* forums [9], where a separate section for time series forecasting has been formed. There, we managed to find some implementations of popular models for the time series forecasting, and the existence of some of them was even acknowledged by Jeremy Howard himself on the forum [10]. Nonetheless, as for today, there is no official, issue-free library of *fastai* used

purely for time series predictions.

Hence, the team had two options:

- use one of the unofficial packages and learn how to use the *fastai* library from the official online course and library's documentation
- focus more on its own model-related experiments

and decided to choose the latter. Nonetheless, it looks like the time series module will be included in the *fastai* library in the future and it might be worth following the development of that tool. It could be potentially used as a baseline solution for the results comparison, achieved in a semi-black-box fashion and requiring less effort when it comes to the implementation of the model.

#### *2.2.3. Attempt utilizing fbprophet library*

Prophet is an open-source forecasting package available for R and Python [11]. It was developed by Facebook's core Data Science team and it takes advantage of non-linear regression and seasonality trends. The idea behind usage of Prophet was the same as with *fastai*, i.e. take a high level tool that is readily available and compare the results to a purpose-built deep learning solution. Prophet has shown promising results but with the caveat that the input can only be a simple time series with no additional attributes (such as CHO and insulin intake in this case). In addition, the fact that evaluation of the results performed by Prophet has not been performed in the same way as in the HDLF, it is difficult to compare the two approaches.

## 3. RESULTS

### 3.1. Evaluation metrics

The performance of the models explored with HDLF is measured with the root mean square error (RMSE), mean absolute error (MAE), both in $mg/dL$, the mean absolute relative difference (MARD) in %, precision, recall, F1 score and finally Clarke and Parkes error grids and time lag, where the latter three measures are domain-specific. The metrics are calculated as:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \hat{x}_i)^2}{N}} \tag{1}$$

$$\text{MARD} = \frac{1}{N} \sum_{i=1}^{N} \frac{|x_i - \hat{x}_i|}{x_i} \tag{2}$$

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} |x_i - \hat{x}_i| \tag{3}$$

where $x_i$ are the CGM measurements, $\hat{x}_i$ are the predictions and $N$ is the total number of data points.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5}$$

where TP stands for true positives, FP stands for false positives and FN stands for false negatives.

$$\text{F1 score} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{6}$$

Both Clarke and Parkes grid are specific to the diabetes domain. They both compare the reference value to the predicted value and based on their position in the respective graphs shown in Figure 3 and Figure 4 categorize the position to one of five areas (A to E) in the error grid. The interpretation of the areas is slightly different between Clarke and Parkes error grid. However, the goal for both of them is to get as many predictions as possible in the area A, which is considered clinically accurate, and as many of the rest to area B which is still considered clinically acceptable. Note that these error grids are also used to measure a performance of a BG meter and the meter is considered to be clinically accurate if it shows at least 95% of the data points in area A [12].

Time lag denotes the time difference between the CGM measurements and the predictions. This is different from the usual meaning of time lag in the context of CGM technology where it denotes the time difference between real BGL and CGM measurements as CGM measures a BG proxy that is lagging behind the real BG. Smaller time lag generally means a better prediction because such algorithm more accurately captures the sudden BG changes.

### 3.2. Final model results

The results of the main model are presented in Table 1. The values are the average over all patients. As discussed before, the model that is tuned for each patient performs better than the generalized one. The most important hyperparameters that made the performance better for the personalized model were the weight decay, learning rate and the number of hidden units. Those hyperparameters were tuned in ranges and from testing the optimizer favoured:

- hidden unit size between 4 and 64 (as powers of 2)
- learning rate between 0.0005 and 0.008
- weight decay between 0.001 and 0.09

| Model | RMSE | MARD | Time lag (min) | Clarke | | Parkes | |
|---|---|---|---|---|---|---|---|
| | | | | A+B | C+D+E | A+B | C+D+E |
| Hedia - Best | 20.027 | 9.97% | 20.45 | 99.57% | 0.43% | —‡ | — |
| Ours - Best Personalized | 20.401 | 10.57% | 20.41 | 97.85% | 2.15% | 99.33% | 0.67% |
| Ours - Best Generalized | 20.492 | 10.66% | 20.49 | 97.83% | 2.17% | 99.26% | 0.74% |
| GluNet[2]† | 19.28 | 8.73% | 8.03 | — | — | — | — |
| CRNNfGP[7]† | 21.07 | — | — | — | — | — | — |

**Table 1**: Results comparison for five different models, including Hedia's top model, our model and two models from research papers. A+B and C+D+E stand for the clinically acceptable and unacceptable areas, respectively, from the Clarke error grid.

†These results are not directly comparable to other results in the table as they are taken from the papers where it is not described in detail how were these values obtained.
‡"—" marks not available values. Hedia best results contained only Clarke results. Results for the last two models are taken from the respective papers. Neither of them measures performance with an error grid and [7] only contains the RMSE value.

Those hyperparameters when tuned for each individual patient outperformed the generalized model across the board. Nonetheless, both models performed similarly with a A+B value close to 98%. Both models have 86% in the A region which is acceptable. For comparison the model from Hedia has 89%. As was mentioned in subsection 3.1, clinically accurate BG meter should have 95% of the measurements in area A. The predictions of both models fall short of that milestone but considering they are predictions and not actual measurements, they do a fairly good job.

For BGL forecasting, it is very important that the model can predict accurately the peaks and valleys because those are the danger zones for the patient. It can be seen in the Figure 5 that those cases are well handled and the forecasting has a good balance between smoothness and jitter.

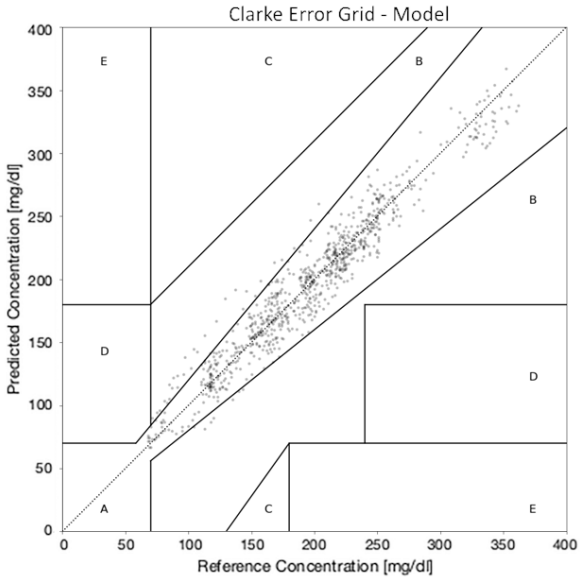



**Fig. 3**: Clarke error grid. Each dot represents a single CGM measurement (x-axis) and its corresponding prediction (y-axis). Notice that a majority of dots fall into the area clinically accurate area A.

## 4. DISCUSSION

As mentioned above, the generalized model performed up to par but not as good as the personalized model. To get the best results, it would make sense to build a model for each individual. Since the training and optimization time does not have to be long in order to get good results, the data from the patients could be reacquired after a period of time, re-train the model and update the optimized model for each individual. Because the generalized model can perform good enough for all patients, it could be used for the first period of data collection. This approach is possible because of the simple model that is proposed hence the re-training and tuning for each patient is possible. This will generate the most accurate forecasting for all patients rather than focusing on building one generalized model for all patients. Hence, since this is a medical solution and each individual is different, such an approach is proposed in order to yield the most value for the customers. Furthermore, after a channel of re-training and optimizing the models has been established, this process can be repeated over time to continuously seek out an improved model.

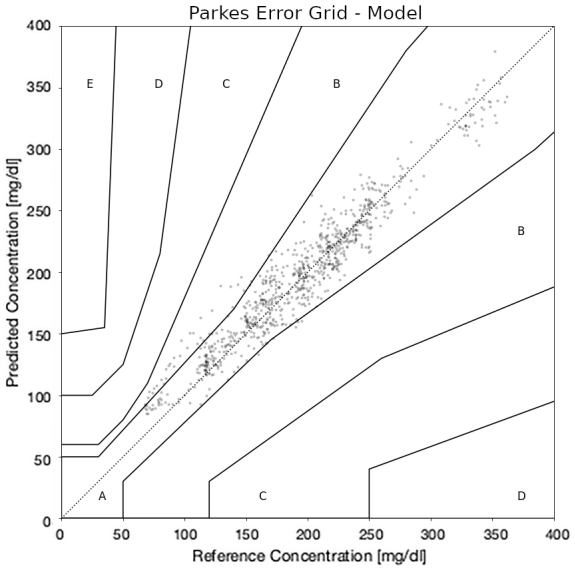



**Fig. 4**: Parkes error grid

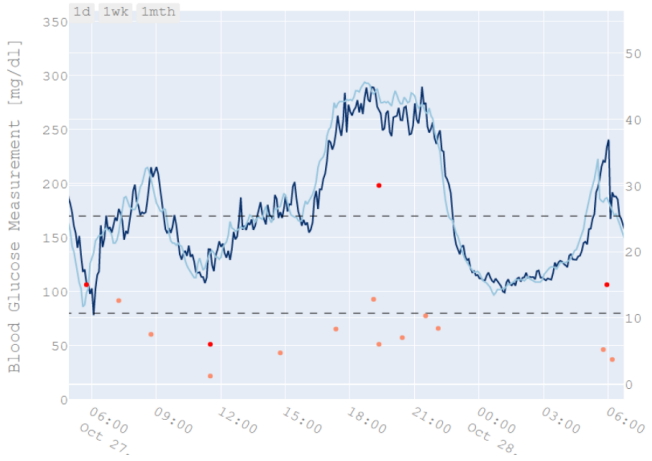



**Fig. 5**: Forecast plot. Dark blue is the prediction and light blue are the true CGM measurements. Notice that the time lag can also be observed - the true CGM measurements are generally before the predictions. The red points mark CHO intake and orange points denote insulin intake. The two dashed lines denote the optimal range for BG (70-180 mg/dL).

An extra benefit of this is that people's habits or glucose patterns might change over time depending on many factors. For this reason an adaptive and re-trainable model makes the most sense.

## 5. CONCLUSION AND FUTURE WORK

Concluding, based on the findings of this report, each individual has different glucose patterns if not unique. This can be supported by the fact that the personalized model outperformed the generalized model across the board. For this reason, a model that is flexible to be optimized to each individual makes the most sense in the real world. In addition, the pattern of an individual could change due to external factors. Since the model does not take into consideration all factors, with the above mentioned approach, the model could adapt to those changes by re-optimizing. This will ensure the quality of the forecasting not only for the first months of use but for the whole period of usage.

Additionally in the future, *fastai* can be revisited once the product matures. It will be interesting to compare what kind of results can a high level deep learning framework provide in comparison with a purpose-built model. In order to achieve a fair comparison between the models, a validation mechanism must be developed in order to test the models in the same manner. Finally, in regards to improving the proposed model's results, an addition could be to introduce daily and weekly seasonality trends as an input to the model. This will act as a helper input that might improve the results. The proposed model already has one helper input, the CGM Delta, in order to support a prediction in the future. In a similar fashion, a seasonality trend could be introduced in order to improve the results further.

## 6. REFERENCES

[1] Pouya Saeedi, Inga Petersohn, Paraskevi Salpea, Belma Malanda, Suvi Karuranga, Nigel Unwin, Stephen Colagiuri, Leonor Guariguata, Ayesha A. Motala, Katherine Ogurtsova, Jonathan E. Shaw, Dominic Bright, and Rhys Williams, "Global and regional diabetes prevalence estimates for 2019 and projections for 2030 and 2045: Results from the international diabetes federation diabetes atlas, 9th edition," *Diabetes Research and Clinical Practice*, vol. 157, pp. 107843, Nov. 2019, DOI: `https://doi.org/10.1016/j.diabres.2019.107843`.

[2] K. Li, C. Liu, T. Zhu, P. Herrero, and P. Georgiou, "Glunet: A deep learning framework for accurate glucose forecasting," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 2, pp. 414–423, 2020, DOI: `https://doi.org/10.1109/JBHI.2019.2931842`.

[3] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica, "Tune: A research platform for distributed model selection and training," *arXiv preprint arXiv:1807.05118*, 2018, `https://arxiv.org/abs/1807.05118`.

[4] C. Marling and Razvan C. Bunescu, "The ohiot1dm dataset for blood glucose level prediction," in *KHD@IJCAI*, 2018, http://smarthealth.cs.ohio.edu/bglp/OhioT1DM-dataset-paper.pdf.

[5] *Pytorch documentation: ELU activation function*, accessed November 2020, https://pytorch.org/docs/stable/generated/torch.nn.ELU.html.

[6] *Pytorch documentation: CELU activation function*, accessed November 2020, https://pytorch.org/docs/stable/generated/torch.nn.CELU.html.

[7] K. Li, J. Daniels, C. Liu, P. Herrero, and P. Georgiou, "Convolutional recurrent neural networks for glucose prediction," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 2, pp. 603–613, 2020, DOI: https://doi.org/10.1109/JBHI.2019.2908488.

[8] Jeremy Howard and Sylvain Gugger, "Fastai: A layered api for deep learning," *Information*, vol. 11, no. 2, pp. 108, 2 2020, DOI: https://doi.org/10.3390/info11020108.

[9] *Fastai forum*, accessed November 2020, https://forums.fast.ai/.

[10] *Fastai forum: TimeSeries post*, 10 2019 (accessed November 2020), https://forums.fast.ai/t/timeseries/55861.

[11] S. Taylor and Benjamin Letham, "Forecasting at scale," *PeerJ Prepr.*, vol. 5, pp. e3190, 2017, doi: https://doi.org/10.7287/peerj.preprints.3190v2.

[12] Andreas Pfützner, David C. Klonoff, Scott Pardo, and Joan L. Parkes, "Technical aspects of the parkes error grid," *Journal of Diabetes Science and Technology*, vol. 7, no. 5, pp. 1275–1281, 2013, DOI: https://doi.org/10.1177/193229681300700517.