**1. Introduction:**

In diabetes care, continuous glucose monitoring (CGM) systems are pivotal for providing real-time glucose readings, which help in maintaining blood glucose levels within the target range. Accurate prediction of glucose levels is crucial due to the severe health risks posed by hypo- and hyperglycemia. Hypoglycemia can lead to symptoms ranging from mild confusion to severe consequences like seizures or loss of consciousness. Hyperglycemia, on the other hand, increases the risk of long-term complications such as cardiovascular diseases, neuropathy, and nephropathy. Maintaining glucose levels within the target range (70-180 mg/dL) is essential to minimize these risks.

The challenge of predicting glucose levels lies in the complex and individual-specific nature of glucose dynamics influenced by various factors such as insulin intake and diet. Current approaches, such as sequence-to-sequence neural networks and recurrent neural networks, offer promising results by leveraging historical glucose data and other relevant inputs to predict future glucose trends. The goal is to develop personalized models that can seamlessly integrate into patients' daily routines, thereby improving their quality of life and health outcomes.

Predictive models utilizing CGM data can alert patients to impending glucose excursions, allowing for proactive adjustments in insulin dosage and dietary intake. Several studies have explored the use of machine learning models for glucose prediction. For instance, Nemat proposed a nested deep ensemble learning framework that improves the precision of blood glucose level predictions by analyzing optimal lag lengths and fusing interconnected time-series data(Khadem et al., 2023). Furthermore, a multi-component deep learning model presented by Zaidi showed high accuracy in multi-step ahead predictions of blood glucose levels(Zaidi et al., 2021). Such predictions can enhance diabetes management and inform timely interventions, thereby improving patient outcomes and reducing the risk of complications associated with diabetes.

**2. Methodology:**

**2.1 Dataset and Preprocessing:**

The dataset is a part of the public dataset built by Aleppo(Aleppo et al., 2017). It was fully joined using “DeviceDtTmDaysFromEnroll”, “DeviceTm” and “PtID”, consists of data collected from adults with well-controlled Type 1 Diabetes. It included metrics such as CGM glucose readings, frequency of blood glucose tests and self-reported food intake. All the attributes included in analysis are shown in Table 1.

Table 1: Variables table

|  |  |  |
| --- | --- | --- |
| Variable name | Type | Description |
| PtID | int | Participant ID |
| DeviceDtTmDaysFromEnroll | int | Device Date – Number of days from enrollment |
| DeviceTm | date time | Device Time |
| GlucoseValue | float | Glucose value (units: mg/dL) |
| CarbInput | Integer | Carbohydrates as inputted into wizard in mg |
| Normal | Integer | Number of units of insulin |

The dataset contains records for 7 patients were age ≥ 18 years, T1D for more than1 year being treated with an insulin pump for at least 3 months. The descriptive statistics for each variable are presented below. Participant 9 didn’t have self-reported food intake. The glucose value and food intake are divided by 100 to achieve normalization. Time variable was transformed into float and then normalized using MinMaxScaler from sklearn. A review of the data reveals no outliers.

Table 2: Distribution of food intake for each participant (100mg)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Participant ID | Mean | Std | Min | Max |
| 2 | 0.11 | 0.24 | 0.00 | 1.62 |
| 3 | 0.44 | 0.33 | 0.00 | 1.60 |
| 5 | 0.20 | 0.12 | 0.00 | 0.70 |
| 7 | 0.31 | 0.30 | 0.00 | 2.20 |
| 8 | 0.27 | 0.21 | 0.00 | 1.20 |
| 9 | NaN | NaN | NaN | NaN |
| 10 | 0.10 | 0.14 | 0.00 | 0.80 |

Table 3: Distribution of dose of insulin for each participant

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Participant ID | Mean | Std | Min | Max |
| 2 | 2.07 | 1.54 | 0.00 | 9.00 |
| 3 | 4.46 | 2.52 | 0.25 | 18.60 |
| 5 | 3.93 | 2.13 | 0.10 | 15.00 |
| 7 | 2.32 | 1.79 | 0.03 | 15.50 |
| 8 | 1.72 | 0.96 | 0.00 | 11.34 |
| 9 | 1.02 | 0.75 | 0.00 | 9.68 |
| 10 | 2.14 | 1.60 | 0.00 | 9.92 |

Table 4: Distribution of glucose value (100 mg/dl).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Participant ID | Mean | Std | Min | Max |
| 2 | 1.27 | 0.48 | 0.39 | 4.11 |
| 3 | 1.66 | 0.65 | 0.39 | 4.01 |
| 5 | 1.49 | 0.52 | 0.39 | 4.01 |
| 7 | 1.66 | 0.64 | 0.39 | 4.01 |
| 8 | 1.59 | 0.65 | 0.39 | 4.01 |
| 9 | 1.55 | 0.48 | 0.39 | 3.52 |
| 10 | 0.97 | 0.21 | 0.39 | 2.88 |

The time proportion in the target range (70-180 mg/dL), in hypoglycaemia (below 70 mg/dL) and in hyperglycaemia (above 180 mg/dL) are shown below. The sum of proportion is not equal to one because there are missing data in glucose value. It can be a result of multiple factors such as: CGM device battery depletion, patient forgetting to apply CGM device and so on.

Table 5: Time proportion of glucose value

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Participant ID | Target range | | Hypoglycemia | Hyperglycemia |
| 2 | 0.54 | 0.03 | | 0.09 |
| 3 | 0.35 | 0.02 | | 0.23 |
| 5 | 0.41 | 0.02 | | 0.13 |
| 7 | 0.38 | 0.03 | | 0.27 |
| 8 | 0.47 | 0.03 | | 0.23 |
| 9 | 0.70 | 0.02 | | 0.27 |
| 10 | 0.35 | 0.01 | | 0.00 |

The forward fill and backward fill were used to do imputation in glucose value and missing data in other columns were filled with 0. Forward fill propagates the last observed non-missing value forward until a new non-missing value is encountered, backward fill is similar to forward fill, but in the opposite direction. They are simple and works well for time-series data. Due to limited number of variables, interpolation and other methods are not applicable.

Figure 1 to 7 are the visualizations of each participant’s time data.

Figure 1.

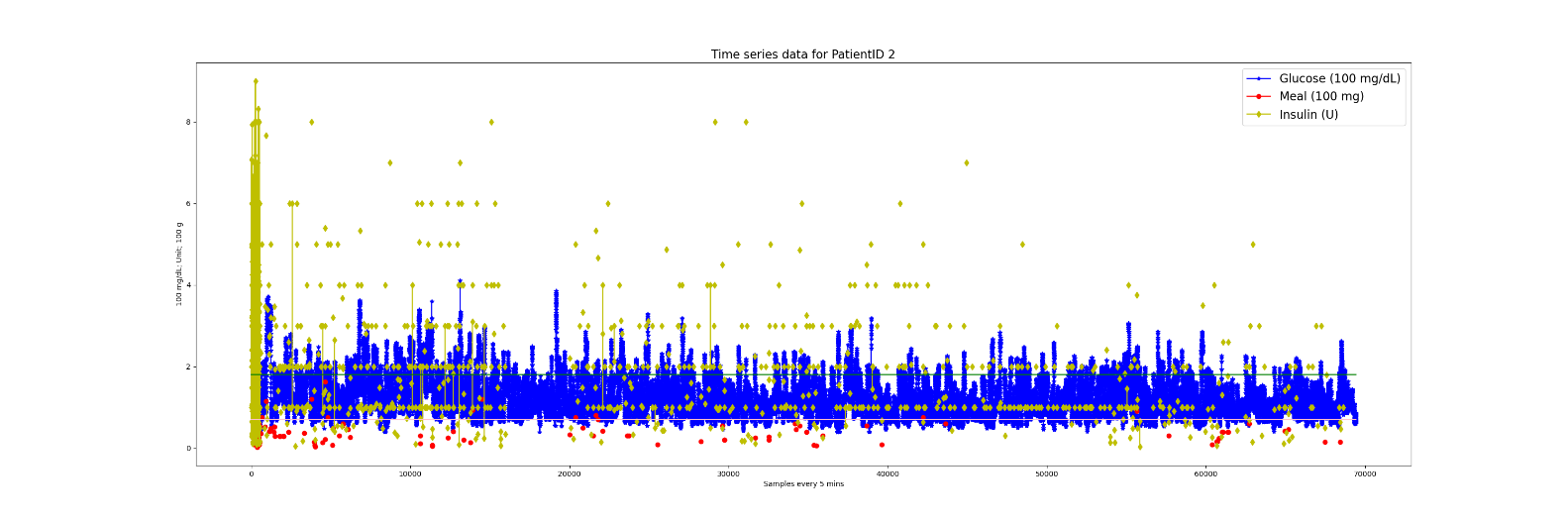


Figure 2.

图表, 散点图

描述已自动生成

Figure 3

图表, 散点图

描述已自动生成

Figure 4

图表

中度可信度描述已自动生成

Figure 5

图表

中度可信度描述已自动生成

Figure 6

图形用户界面

描述已自动生成

Figure 7

图表, 散点图

描述已自动生成

The dataset was divided into 70% training dataset, 10% validation dataset, 20% testing dataset, the prediction window is 30 minutes and sliding window size is 2 hours and step size of 1 (5 minutes).

**2.2 Model development:**

**2.2.1 Support Vector Machines (SVM):**

SVMs are supervised learning models used for classification and regression tasks. The goal of SVM is to find a hyperplane in an N-dimensional space (N — the number of features) that distinctly classifies the data points. The best hyperplane is the one that has the largest distance to the nearest training data points of any class (functional margin), as this can help reduce generalization error(Trevor Hastie, 2009).

Given a training dataset (xn, yn) where xi are the data points and yi are the class labels, the decision function for a SVM is defined as:

(1)

sgn is the sign function that returns 1 if the argument is positive and 0 if it is negative.

are the Lagrange multipliers obtained from solving the SVM's dual optimization problem.

K(xi ,x) is the kernel function, which computes the inner product of vectors xi and x in the transformed feature space.

b is the bias term, which shifts the decision boundary away from the origin.

Radial Basis Function (RBF) Kernel:

(2)

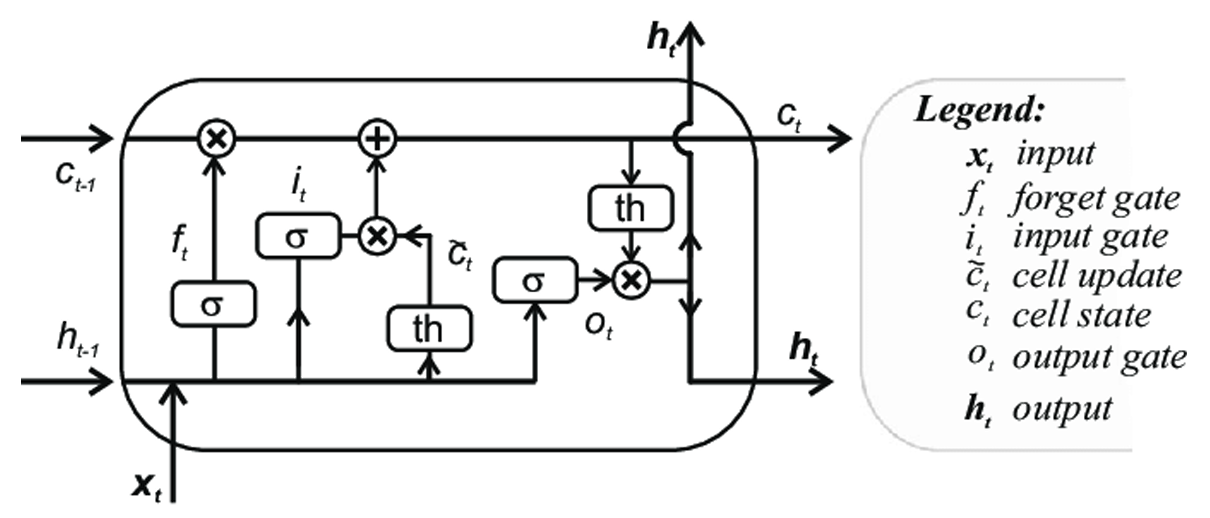
Table 6: Pseudo code for SVM

|  |
| --- |
| Choose an SVM Kernel:  Based on the dataset, decide which kernel function to use:  Considered training data, use a radial basis function (RBF) kernel.    SVM Model Initialization:  Initialize the SVM classifier with the chosen kernel.  Set the initial values for SVM hyperparameters:  Regularization parameter (C) to control the trade-off between smooth decision boundary and classifying training points correctly.  Kernel-specific parameters (like gamma for the RBF kernel).    Model Training:  Train the SVM classifier using the training dataset:  The algorithm learns the weights that define the decision boundary hyperplane.  The algorithm identifies the support vectors that are critical data points defining the decision boundary. |

**2.2.2. Long-Short Term Memory (LSTM):**

Long-Short Term Memory (LSTM) networks are a leading accuracy in complex time series prediction tasks, by leveraging its capability to bridge long-term dependencies in excess of 1000 time steps(Hochreiter and Schmidhuber, 1997).

Figure 8: Graphic illustration of LSTM cell



LSTM cells hold two types of memory: short and long. Hidden States (ht) are used throughout Recurrent Neural Networks (RNN) to store immediate previous events and are constantly majorly altered by a series of gates. The internal Cell State stores long term dependencies, with only minor linear interactions throughout the forward pass. This enhances RNN performance, which is inhibited by the vanishing gradient problem. Both are represented as vectors of predefined size, with larger vectors increasing model complexity(Sherratt et al., 2021, Xia et al., 2019).

Both the cell and hidden states are fed back to the input of the cell, alongside the next time step and the calculations are repeated until the whole sequence is processed. Once the final time step is reached the terminal cell state is passed to a fully connected output layer, which subsequently produces the final output.

The forget gate (ft) is a sigmoid layer selecting what information to discard from the cell state. The input (It) and output (Ot) gates transform the previous hidden state (ht-1), using a sigmoid activation function, to incorporate into cell state (Ct) and output hidden state (ht) respectively.

Bidirectional LSTM (BiLSTM) architecture is similar to LSTM, however the sequence is processed in both the forward and backwards direction, which requires the entire sequence to be available at once.

The training, validation, and test sets were loaded using PyTorch's DataLoader, with a batch size of 128. Creation of the LSTM model was performed in an object-oriented approach permitted by the PyTorch framework in Python.

A bidirectional LSTM layer with a hidden size of 50 was designed. After the LSTM layer, the output passes through a series of fully connected (Linear) layers with ReLU activations. The learning rate is set to 0.001 and weight decay is set to 0.0001. The Adam optimizer was used to update the model parameters.

**2.3 Hyperparameter tuning:**

Hyperparameters were tuned using GridSearchCV() from sklearn package, the scoring includes mean absolute error and mean squared error.

The Mean Squared Error (MSE) loss function was chosen as the cost function. The model's performance was validated on the validation set at the end of each epoch. The model achieving the lowest validation loss was saved as the best model. The number of epochs is set to 20.

**2.4 Model evaluation:**

The evaluation will be conducted using a 5-fold cross-validation strategy (20% of the sequences being used to test and 70% to train), in addition to a random sampler, with a predefined seed to ensure repeatability.

Accuracy of the models will be evaluated with an array of metrics, for thorough comparative inspection of different loss types1.

Root Mean Squared Error (RMSE) quantifies the proximity of predictions to the validation set. The sklearn Mean Squared Error (MSE) and Mean Absolute Percentage Error loss function used and then averaged over the number of batches in a training epoch.

**3.Results:**

In this study, the performance of two machine learning models, Support Vector Machine (SVM) and Long Short-Term Memory (LSTM) was evaluated in predicting glucose levels for seven participants. The models' performance was assessed using two metrics: Root Mean Squared Error (RMSE) and Mean Absolute Relative Difference (MARD).

Overall, the LSTM model consistently outperformed the SVM model across most participants, showing lower RMSE and MARD values. This suggests that LSTM, with its capability to capture temporal dependencies, may be better suited for predicting glucose levels from continuous glucose monitoring data. The performance varied among participants, indicating that individual variability plays a significant role in model accuracy. The results are summarized in Table 7.

Table 7: Comparison of RMSE and MARD for each participant

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Participant ID | Model | | RMSE | MARD |
| 2 | SVM | 15.45 | | 9.71 |
| LSTM | 15.34 | | 9.17 |
| 3 | SVM | 28.19 | | 13.73 |
| LSTM | 26.27 | | 14.27 |
| 5 | SVM | 24.85 | | 11.22 |
| LSTM | 19.57 | | 9.78 |
| 7 | SVM | 23.30 | | 11.94 |
| LSTM | 21.44 | | 10.76 |
| 8 | SVM | 20.71 | | 11.21 |
| LSTM | 19.60 | | 10.81 |
| 9 | SVM | 22.43 | | 11.59 |
| LSTM | 16.01 | | 6.89 |
| 10 | SVM | 11.49 | | 7.45 |
| LSTM | 11.36 | | 7.42 |

The LSTM model was trained for 20 epochs, and the training and validation loss values were shown in Figure 9. Both the training loss and validation loss show a decreasing trend over the epochs. The training loss starts relatively high (0.16499) and steadily decreases to 0.03585 by the 20th epoch. The validation loss also shows a consistent decrease from 0.03319 to 0.02686.

Figure 9: Model loss

图表

描述已自动生成

The SVM model was evaluated using the Mean Squared Error metric, the optimal hyperparameters were determined using GridSearchCV, which systematically works through multiple combinations of parameter tunes, cross-validating as it goes to determine which tune gives the best performance. The best combination of parameters that minimized this error was shown in Table 8.

Table 8: Optimal hyperparameters for SVM

|  |  |  |
| --- | --- | --- |
| Scoring | Best Score | Best Parameters |
| Mean squared error | 238.70 | C=100， gamma = 0.01, kernel = ‘rbf’ |

The experiments were performed on a PC equipped with an Intel i7-13700H processor (2.4 GHz base clock speed), 16 GB RAM. The operating system was Windows 11, version 23H2. The machine learning models were implemented using Python 3.11.4 with scikit-learn version 1.4.0. The PyTorch version is 2.2.0.

The number of cross validations is 5.

The comparisons of the true data, predictions from SVM model and LSTM model are shown below:

Figure 10: Prediction comparison for participant 2

**图表, 折线图, 直方图

描述已自动生成**

Figure 11: Prediction comparison for participant 3

**图表, 折线图, 直方图

描述已自动生成**

Figure 12: Prediction comparison for participant 5

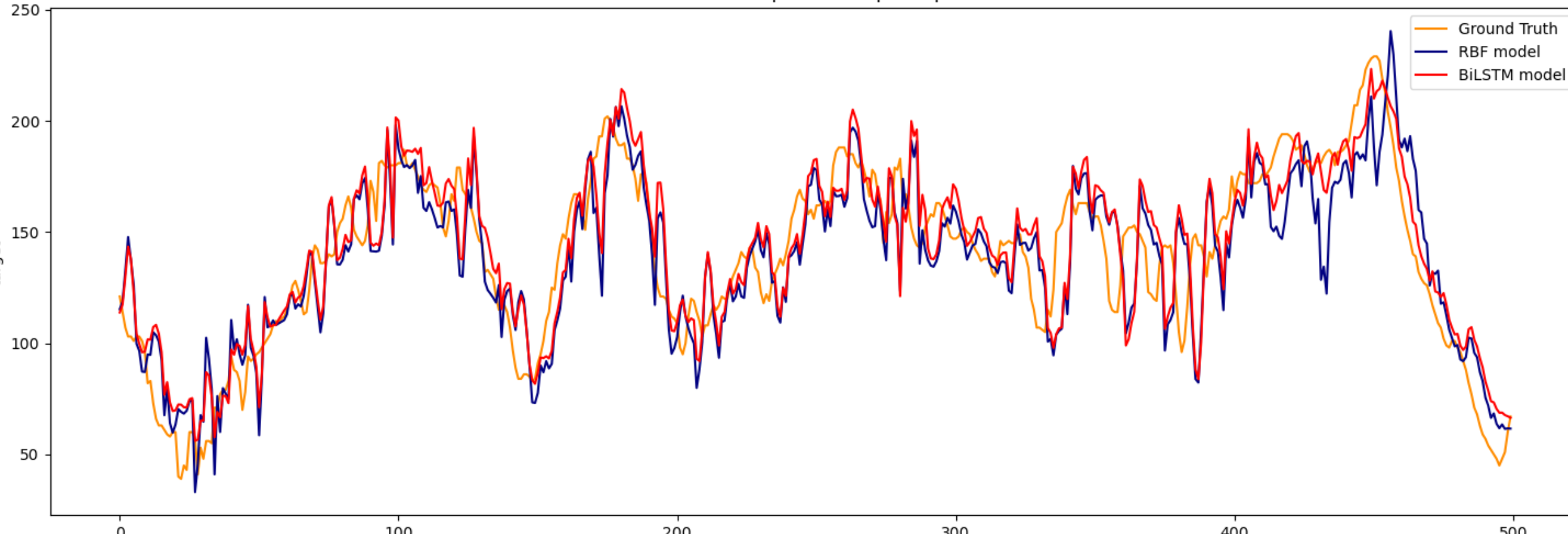
****

Figure 13: Prediction comparison for participant 7

图表, 折线图, 直方图

描述已自动生成

Figure 14: Prediction comparison for participant 8

**图表, 折线图, 直方图

描述已自动生成**

Figure 15: Prediction comparison for participant 9

**图表, 直方图

描述已自动生成**

Figure 16: Prediction comparison for participant 10

**图片包含 链, 项链

描述已自动生成**

Both the LSTM model and SVM model are well-trained and generalizes effectively to unseen data.

**4. Discussion:**

Future investigations can be conducted with addition of relevant patient characteristics such as weight, age and annual income directly to the model to provide further context.

An opportunity for deployment of the LSTM model in edge devices could arise, due to its small size and low energy requirements for inference. Sensitive medical data could be stored locally on the device and predictions computed locally(Sun et al., 2020).

When evaluating the model for predicting glucose value, there are several metrics that can be considered beyond RMSE and MARD, such as Mean Absolute Error (MAE) and R². MAE can be used to measure the average magnitude of the errors in a set of predictions, without considering their direction. R² indicates the proportion of the variance in the dependent variable that is predictable from the independent variables. It provides an indication of the goodness of fit of the model.

Glucose dynamics can vary significantly between individuals due to factors such as differences in metabolism, lifestyle, diet, and insulin sensitivity . This variability can make it challenging for a model trained on pooled data to accurately predict glucose levels for a specific individual(Khadem et al., 2023, Herrero et al., 2022).

While general patterns can be beneficial, there is a risk that the model may overfit to these patterns and fail to capture unique individual behaviours. It can lead to suboptimal performance when the model is applied to a specific patient whose data patterns differ from the training set .

Including data from multiple patients increases the complexity of the model, which can make it harder to interpret and fine-tune. That’s why personalized models are used under this study.

Although population models that trained on data from multiple patients can learn generalized patterns and improve its robustness to variations. And developing a single model for the entire population can be more resource-efficient than creating and maintaining separate models for each individual. It allows leveraging pooled computational resources and avoids redundancy. Lately, there are emerging research indicate that a hybrid model, which combine population-based pre-training with personalized fine-tuning, often yield the best results in terms of prediction accuracy and robustness. This method combines the robustness and generalization capability of population models with the precision and adaptability of personalized models, leading to improved prediction accuracy.

**6. Reference:**

1. ALEPPO, G., RUEDY, K. J., RIDDLESWORTH, T. D., KRUGER, D. F., PETERS, A. L., HIRSCH, I., BERGENSTAL, R. M., TOSCHI, E., AHMANN, A. J., SHAH, V. N., RICKELS, M. R., BODE, B. W., PHILIS-TSIMIKAS, A., POP-BUSUI, R., RODRIGUEZ, H., EYTH, E., BHARGAVA, A., KOLLMAN, C., BECK, R. W. & GROUP, R.-B. S. 2017. REPLACE-BG: A Randomized Trial Comparing Continuous Glucose Monitoring With and Without Routine Blood Glucose Monitoring in Adults With Well-Controlled Type 1 Diabetes. Diabetes Care, 40, 538-545.

2. HERRERO, P., REDDY, M., GEORGIOU, P. & OLIVER, N. S. 2022. Identifying Continuous Glucose Monitoring Data Using Machine Learning. Diabetes Technol Ther, 24, 403-408.

3. HOCHREITER, S. & SCHMIDHUBER, J. 1997. Long short-term memory. Neural Computation, 9, 1735-1780.

4. KHADEM, H., NEMAT, H., ELLIOTT, J. & BENAISSA, M. 2023. Blood Glucose Level Time Series Forecasting: Nested Deep Ensemble Learning Lag Fusion. Bioengineering (Basel), 10.

5. SHERRATT, F., PLUMMER, A. & IRAVANI, P. 2021. Understanding LSTM Network Behaviour of IMU-Based Locomotion Mode Recognition for Applications in Prostheses and Wearables. Sensors, 21.

6. SUN, L. F., JIANG, X., REN, H. X. & GUO, Y. 2020. Edge-Cloud Computing and Artificial Intelligence in Internet of Medical Things: Architecture, Technology and Application. Ieee Access, 8, 101079-101092.

7. TREVOR HASTIE, R. T., JEROME FRIEDMAN 2009. The Elements of Statistical Learning Data Mining, Inference, and Prediction, Second Edition, Springer New York, NY.

8. XIA, J., PAN, S., ZHU, M., CAI, G., YAN, M., SU, Q., YAN, J. & NING, G. 2019. A Long Short-Term Memory Ensemble Approach for Improving the Outcome Prediction in Intensive Care Unit. Comput Math Methods Med, 2019, 8152713.

9. ZAIDI, S. M. A., CHANDOLA, V., IBRAHIM, M., ROMANSKI, B., MASTRANDREA, L. D. & SINGH, T. 2021. Multi-step ahead predictive model for blood glucose concentrations of type-1 diabetic patients. Sci Rep, 11, 24332.