# Evaluation of Optimally Tuned K-Nearest Neighbors for 30-Minute Blood Glucose Prediction in Type 1 Diabetes Using OhioT1DM Dataset

Yacine Hachi[1*], Soraya Tighidet[1], Kamal Amroun[1], Meriem Djouadi[2]
[1]Limed Laboratory, Faculty of Exact Sciences, University of Bejaia, 06000 Bejaia, Algeria
[2]Department of Computer Science, Faculty of Exact Sciences, Echahid Hamma Lakhdar University, P. O. B. 789, 39000 El Oued, Algeria
E-mail: yacine.hachi@univ-bejaia.dz, soraya.tighidet@univ-bejaia.dz, kamal.amroun@univ-bejaia.dz, djouadi-meriem@univ-eloued.dz
*Corresponding author

**Keywords:** diabetes, predict, blood glucose levels, hypoglycemia, hyperglycemia, K-Nearest Neighbors, machine learning

**Received:** February 18, 2025

*Diabetes is a long-term chronic medical condition with the potential to evolve into a global healthcare crisis, glycemic control is fundamental for the effective management of diabetes and the prevention of its associated complications. Forecasting future blood glucose levels (BGLs) for diabetic patients can help them avoid serious health problems. This study investigates the application of the KNN regression algorithm to predict future (BGLs), utilizing historical blood glucose measurements from twelve patients (six patients from the Ohio dataset version 2018 and six patients from the Ohio dataset version 2020) as the only input feature. Our proposed approach employed a methodology that utilized historical measurements to train predictive models. Specifically, we leveraged the following historical data points - (BGLs) at 4-hours, 8-hours, 12-hours, 16-hours, 20-hours, and 24-hours intervals - as input features to predict (BGLs) 30 minutes into the future. This study explores the impact of varying parameters of the KNN algorithm, such as the K value= [2,3,5,7,11], weights= ['uniform', 'distance'] and distance metric= ['euclidean', 'manhattan', 'minkowski'], on the performance of the model. Furthermore, we compared the obtained results of the KNN algorithm with other machine learning methods, including linear regression, Random Forests, Support Vector Machines, CatBoostRegressor, LightGBM, XGBoost, artificial neural networks and previous studies. Among these, KNN yielded the best results with optimal hyperparameters (k=2, Weights='distance', Metric='manhattan') in the tow version of datasets OhioT1DM V2018 and OhioT1DM V2020. The OhioT1DM V2018 dataset yielded optimal performance with an RMSE of 5.09 ± 0.91 mg/dl using a 24-hour window size, and an MAE of 2.42 ± 0.34 mg/dl with a 12-hour window size. For the OhioT1DM V2020 dataset, the best results were an RMSE of 5.56 ± 1.14 mg/dl with a 12-hour window size, and an MAE of 2.47 ± 0.34 mg/dl achieved using an 8-hour window size. This research confirms that KNN algorithm with optimal hyperparameters (k=2, Weights='distance', Metric='manhattan') can effectively predict blood glucose events, which will help prevent and reduce the occurrence of serious complications such as hypoglycemia and hyperglycemia.*

*Povzetek: Študija optimizira algoritem KNN za 30-minutno napovedovanje ravni glukoze v krvi (BGL) pri sladkorni bolezni tipa 1 z uporabo podatkov OhioT1DM. Optimalni hiperparametri (k=2, Weights='distance', Metric='manhattan') so dosegli najboljše rezultate.*

## 1 Introduction

Diabetes mellitus has emerged as one of the most pressing global health concerns, with over 463 million individuals affected in 2019, projections indicate that this figure is poised to escalate further, reaching an estimated 700 million by the year 2045 [1]. The treatment of diabetes type1, which primarily relies on the administration of external insulin, necessitates the frequent assessment of blood glucose levels, currently this monitoring is facilitated by continuous glucose monitoring devices (CGM), which enable the collection and display of glucose concentrations in an almost continuous manner for multiple days [2]. In certain scenarios, CGM interface with an insulin pump, which mimics the natural functioning of the pancreas by administering small, on-demand doses of insulin, consequently over the past decade, researchers have dedicated their efforts to developing machine learning-based algorithms that can accurately predict future blood glucose levels [3]. Many previous studies have proposed numerous predictive algorithms. The

researchers in [4] compared three different models; an autoregressive model that utilized only glucose data, an autoregressive model that incorporated external insulin information, and an artificial neural network (ANN) that leveraged both glucose and insulin data. Furthermore, online adaptive models were employed to account for the inherent intra-individual and inter-individual variability present in the diabetic population. Hamdi et al. [5] proposed utilizing solely CGM data to forecast BGLs independently of other variables. To substantiate this approach, they investigated the application of (SVR) and differential evolution algorithms (DE). The authors in [6] presented a deep learning model utilizing a dilated recurrent neural network (DRNN) architecture to generate 30-minute forecasts of prospective glucose levels. Additionally, they leveraged a transfer learning approach that incorporated dilation to harness data from multiple participants. Dudukcu. H.V et al [7] proposed several advanced neural network architectures, constituting (LSTM), WaveNet, and Gated Recurrent Units (GRU). The hyperparameters of these models were tuned to optimize their operational efficiency. The authors in [8] proposed an autonomous deep learning model for personalized forecasting of multivariate BGLs. The proposed autonomous channel network acquires representations from input variables with appropriate sequence lengths and sampling periods, drawing on domain knowledge of the time-dependent relationships between the variables. Shuvo. M. et al. [9] described a neural network architecture comprising shared and clustered hidden layers. The shared hidden layers, composed of two stacked long short-term memory layers, learned generalized features across all data samples. In contrast, the clustered hidden layers, consisting of two dense layers, adapted to the gender-specific variations present in the data. This study presents a predictive model that utilizes the KNN algorithm to forecast blood glucose levels within a 30-minute in the future. The effectiveness of this model is evaluated through the root mean square error rate (RMSE) and mean absolute error (MAE). In addition, many experiments were conducted to determine the optimal values of K, weights and the distance measure. We also experimented with various window sizes in order to minimize the error and enhance the model's performance.

The fundamental question addressed in this article, to be explored further in the discussion section, is whether a straightforward model like KNN can achieve superior performance compared to advanced models such as deep learning in predicting blood glucose levels for diabetic patients within a short prediction horizon (30-minutes) using only historical CGM data?

The reminder of the paper is organized as follows. Section 2 data and preprocessing. Our methodology is detailed in Section 3. Experimental results are presented and discussed in Section 4. Section 5 presents the limitations. Finally, section 6 provides a conclusion of the paper.

# 2 Data and preprocessing

The study employed the two datasets, OhioT1DM in 2018 and OhioT1DM in 2020 [10], which both contain training and testing data. These datasets provide information on twelve individuals with type 1 diabetes over an eight-week timeframe, including details such as glucose levels, finger stick readings, bolus doses, basal rates, interim basal rates, meal consumption, exercise routine, and basal heart rate. The data for each patient is segregated into distinct training and testing subsets within the overall dataset. Table 1 presents the number of training and test samples for each patient, along with their respective split ratios.

Table 1: Number of training and test samples for each patient

| **OhioT1DM V2018** | | | | |
|---|---|---|---|---|
| **ID_Patient** | **Gender** | **Age** | **Training Examples 80 %** | **Test Examples 20%** |
| 559 | female | 40–60 | 10796 | 2514 |
| 563 | male | 40–60 | 12124 | 2570 |
| 570 | male | 40–60 | 10982 | 2745 |
| 575 | female | 40–60 | 11866 | 2590 |
| 588 | female | 40–60 | 12640 | 2791 |
| 591 | female | 40–60 | 10847 | 2760 |
| **OhioT1DM V2020** | | | | |
| 540 | male | 20–40 | 11947 | 2884 |
| 544 | male | 40–60 | 10623 | 2704 |
| 552 | male | 20–40 | 9080 | 2352 |
| 567 | female | 20–40 | 10858 | 2377 |
| 584 | male | 40–60 | 12150 | 2653 |
| 596 | male | 60–80 | 10877 | 2731 |

This study utilized a sliding window technique to transform time-series data into an input matrix and output vector dependent solely on BGLs measured by CGM every 5 minutes. We are focusing only on blood glucose levels as input values for all predictive models, excluding factors such as diet, physical activity, stress, or drug treatments. This approach was adopted due to the challenges in accurately measuring and quantifying these other variables in real-world scenarios. Moreover, additional factors could be considered to enhance the model's accuracy, but these may impose a burden on the patient. Additionally, this study aims to reduce the number of input features while improving model accuracy. We conducted several experiments to explore the impact of window size. Specifically, we used:

- P = 48 previous data points (4-*h*) to train the model and forecast the BGL for the next F = 6 data points (30-minutes).

- P = 96 previous data points (8-*h*) to train the model and forecast the BGL for the next F = 6 data points (30-minutes).
- P = 144 previous data points (12-*h*) to train the model and forecast the BGL for the next F = 6 data points (30-minutes).
- P = 192 previous data points (16-*h*) to train the model and forecast the BGL for the next F = 6 data points (30-minutes).
- P = 240 previous data points (20-*h*) to train the model and forecast the BGL for the next F = 6 data points (30-minutes).
- P = 288 previous data points (24-*h*) to train the model and forecast the BGL for the next F = 6 data points (30-minutes).

The Figure 1 depicts the illustration of the sliding window technique for window size = 4-hours.

# 3 Methodology

## 3.1 K- Nearest Neighbors (KNN) Algorithm

The K-Nearest Neighbors algorithm is a non-parametric supervised learning technique that can be applied to both classification and regression problems. The fundamental operation of this algorithm is predicated on the concept of similarity between the data points within the dataset [11].

The accuracy and efficacy of the K-Nearest Neighbors regression model are primarily influenced by the choice of the K parameter as well as the distance metric employed. A variety of techniques can be employed to measure the similarity between data points. In our investigation, we utilized the Euclidean formula (1), Manhattan formula (2), and Minkowski formula (3) distance metrics to quantify proximity.

The Euclidean distance between two points X and Y can be calculated using the formula (1):

$$D(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (1)$$

The Manhattan distance between two points A and B can be calculated using the formula (2):

$$D(A,B) = \sum_{i=1}^{m}|A_i - B_i| \quad (2)$$

The Minkowski distance combines the Euclidean and Manhattan distance metrics to quantify the separation between two data points X and Y through a mathematical expression formula (3):

$$D(X,Y) = (\sum_{i=1}^{m}|X_i - Y_i|^X)^{\frac{1}{X}} \quad (3)$$

Additionally, the selection of the K value significantly impacts the performance of the K-Nearest Neighbors algorithm. For this reason, our study undertook multiple experiments to determine the optimal K value, as well as the most suitable distance metric to apply.

The key steps involved in the KNN regression procedure employed in our study are as follows:

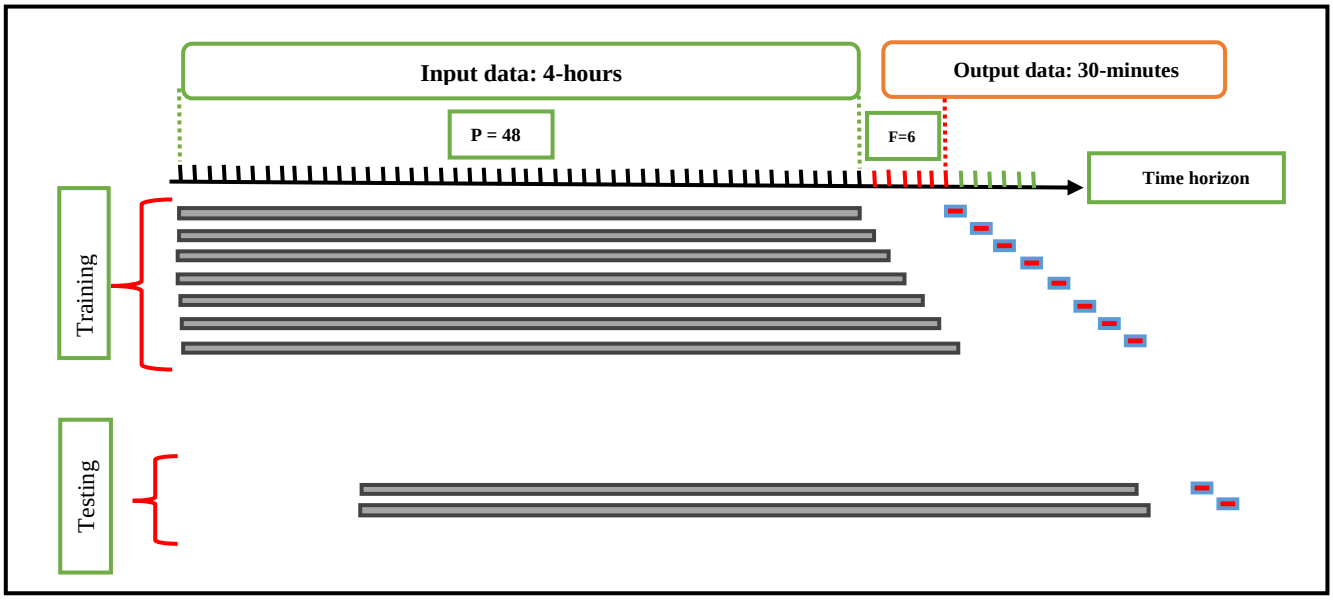


Figure 1: Sliding window approach with a window size of 4 hours

**Procedure**

**Data:** CGM from OhioT1DM

**Result:** Best minimum RMSE, MAE and the associated hyperparameter configuration.

**1.** Divide the dataset into training and test sets (X_train, X_test, y_train, y_test).

**2.** Initialize the KNN model

**3.** Hyperparameter values to be evaluated { # Define the parameter grid

**N_neighbors** = [2,3,5,7,11]

**Weights** = ['uniform', 'distance']

**Metric** = ['euclidean', 'manhattan', 'minkowski']

}.

**4.** kfold = KFold(n_splits=5) # Define 5-fold cross-validation

**5.** GridSearchCV(knn, hyperparameter values, cv=kfold) # Perform grid search with cross-validation for hyperparameter tuning

**6.** Fit the hyperparameter values on the training data (X_train, y_train).

**7.** Select the optimal model from the hyperparameter values.

**8.** Predict the test set using the best model.

**9.** Calculate **RMSE, MAE** for the predictions.

**10.** Report the optimal **RMSE, MAE** and the associated hyperparameter configuration.

End.

This study evaluated the performance of the K-Nearest Neighbors algorithm on the OhioT1DM Diabetes dataset through an experimental analysis. The aim of this investigation was to determine the optimal values for the number of neighbors (K), the Weights and the distance metric that would maximize the performance of the KNN algorithm.

The hyperparameters used in other machine learning models, including linear regression (copy_X= True,fit_intercept=True,n_jobs=None, positive=False), SVR(kernel='rbf', C=100, gamma=0.1, epsilon=0.1), CatBoostRegressor( 'iterations': 1000, 'learning_rate': 0.01,'depth':6,'random_state':42),Lightgbm('n_estimators':1000, 'learning_rate': 0.01, 'num_leaves': 31, 'random_state': 42), ANN with 3 layers((128, activation='relu'), Dropout(0.3), (64,activation='relu'), Dropout(0.3),1,optimizer='adam',loss='mean_squared_error')) and (epochs=100, batch_size=128), XGBoost( 'learning_rate':0.01,'n_estimators':1000,'random_state': 42,num_boost_round=1000),RandomForest(n_estimators=100,max_depth=None,min_samples_split=2,min_samples_leaf=1, bootstrap=True). The schematic diagram presented in Figure 2 illustrates the methodology of our approach used in the predictive models. Forecasting models were implemented in Python (version 3.9.12) using CPU, with scikit-learn (version 1.6.1), NumPy (version 1.24.3), Pandas (version 1.4.2), and TensorFlow (version 2.13.0).

# 4 Results

The metrics employed in the evaluation of the efficiency and accuracy of the algorithms were as follows:

## 4.1 Root Mean Square Error (RMSE) and Mean Absolutely Error (MAE)

The RMSE, a widely employed metric, measures the average magnitude of the errors in a predictive model, can be characterized as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)} \quad (4)$$

Where the predicted output is denoted as $\hat{y}_i$ and the true output is denoted as $y_i$. The root mean square error metric offers several desirable properties, including a readily defined gradient, intuitive interpretation, and the ability to transform the error back to the original scale through the square root operation [12].

The mean absolute error is quantified as:

$$MAE = \frac{1}{n}\sum_{n=1}^{n}|(\hat{y}_i - y_i)| \quad (5)$$

This error measure is simple to formulate and exhibits a degree of insensitivity to outliers, we used MAE as a second metric for assessing the accuracy of our regression model.

## 4.2 Comparison of our approaches with the relevant prior work in the literature

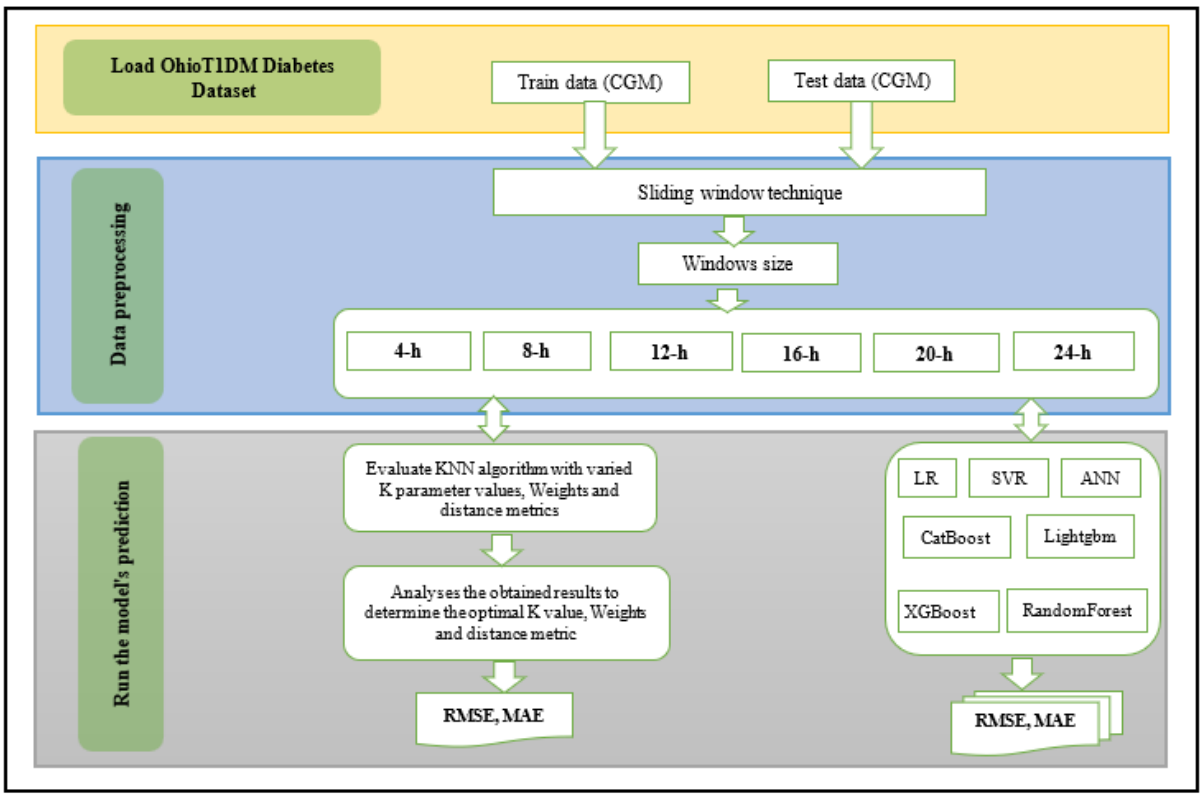


Figure 2: Flowchart of our approach used in the predictive models.

Table 2: Mean with standard deviation (Mean ± Std) of RMSE, MAE for Ohio T1DM Dataset version 2018 in different windows size

| Name of the model | window size | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *4-h* | | *8-h* | | *12-h* | | *16-h* | | *20-h* | | *24-h* | |
| | **RMSE** | **MAE** | **RMSE** | **MAE** | **RMSE** | **MAE** | **RMSE** | **MAE** | **RMSE** | **MAE** | **RMSE** | **MAE** |
| LR | 23.63 ± 3.89 * | 16.18 ± 2.60 ** | 23.71 ± 3.89 ** | 16.26 ± 2.61 ** | 23.73 ± 3.87 ** | 16.29 ± 2.61 ** | 23.81 ± 3.94 ** | 16.38 ± 2.67 ** | 23.79 ± 3.84 ** | 16.40 ± 2.63 ** | 23.86 ± 3.82 ** | 16.47 ± 2.65 ** |
| SVR | 58.62 ± 7.87 ** | 47.19 ± 6.55 ** | 58.64 ± 7.83 ** | 47.21 ± 6.51 ** | 58.66± 7.82 *** | 47.22 ± 6.48 ** | 58.63 ± 7.82 ** | 47.19 ± 6.49 ** | 58.56 ± 7.83 ** | 47.12 ± 6.50 ** | 58.51 ± 7.89 ** | 47.06 ± 6.54 ** |
| CatBoostRegressor | 22.31 ± 3.49 * | 15.66 ± 2.36 ** | 21.86 ± 3.37 ** | 15.40 ± 2.28 ** | 21.53 ± 3.20 ** | 15.22 ± 2.22 ** | 21.40 ± 3.24 ** | 15.14 ± 2.27 ** | 21.16 ± 3.23 ** | 15.00 ± 2.27 ** | 20.95 ± 3.15 ** | 14.88 ± 2.23 ** |
| Lightgbm | 20.11 ± 3.16 * | 13.95 ± 2.11 ** | 18.72 ± 2.84 ** | 13.01 ± 1.90 ** | 17.84 ± 2.65 ** | 12.44 ± 1.79 ** | 17.28 ± 2.64 * | 12.05 ± 1.79 ** | 16.82 ± 2.63 * | 11.75 ± 1.79 ** | 16.30 ± 2.44 * | 11.40 ± 1.65 ** |
| ANN | 33.36 ± 4.13 ** | 26.18 ± 3.26 ** | 39.02 ± 3.47*** | 31.83 ± 2.71*** | 42.35 ± 3.61*** | 34.97 ± 4.61 ** | 46.09 ± 5.68*** | 38.60 ± 6.24 ** | 48.30 ± 5.10*** | 40.64 ± 5.04*** | 49.25 ± 3.85*** | 41.40 ± 3.38*** |
| XGBoost | 19.95 ± 3.17 * | 13.78 ± 2.09 ** | 18.60 ± 2.84 ** | 12.86 ± 1.88 ** | 17.70 ± 2.68 ** | 12.27 ± 1.78 ** | 17.15 ± 2.64 * | 11.88 ± 1.78 ** | 16.84 ± 2.65 * | 11.66 ± 1.81 ** | 16.35 ± 2.54 * | 11.34 ± 1.71 ** |
| RandomForest | 18.89 ± 3.17 * | 12.23 ± 2.01 * | 17.50 ± 2.92 * | 11.00 ± 1.79 ** | 16.73 ± 2.73 * | 10.37 ± 1.68 ** | 16.33 ± 2.69 * | 10.00 ± 1.67 * | 15.99 ± 2.75 * | 9.77 ± 1.72 * | 15.65 ± 2.69 * | 9.50 ± 1.65 * |
| **KNN with best hyperparameter (**K=2, Weights='distance'**,** Metric='manhattan') | 7.78 ± 1.63 | 3.30 ± 0.56 | 5.41 ± 1.10 | 2.45 ± 0.35 | 5.30 ± 1.21 | **2.42 ± 0.34** | 5.29 ± 0.99 | 2.45 ± 0.38 | 5.22 ± 1.00 | 2.48 ± 0.35 | **5.09 ± 0.91** | 2.47 ± 0.39 |

*p ≤ 0.0001 **p ≤ 0.00001 ***p ≤ 0.000001.

Table 3: Mean with standard deviation (Mean ± Std) of RMSE, MAE for Ohio T1DM Dataset version 2020 in different windows size

| Name of the model | window size | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *4-h* | | *8-h* | | *12-h* | | *16-h* | | *20-h* | | *24-h* | |
| | **RMSE** | **MAE** | **RMSE** | **MAE** | **RMSE** | **MAE** | **RMSE** | **MAE** | **RMSE** | **MAE** | **RMSE** | **MAE** |
| LR | 24.64 ± 3.96 * | 17.04 ± 2.57 ** | 24.70 ± 3.97 * | 17.10 ± 2.60 ** | 24.77 ± 3.99 * | 17.15 ± 2.60 ** | 24.87 ± 4.02 * | 17.23 ± 2.60 ** | 24.92 ± 3.99 * | 17.29 ± 2.59 ** | 24.92 ± 4.00 * | 17.33 ± 2.59 ** |
| SVR | 57.56 ± 5.06*** | 45.59 ± 3.68*** | 57.73 ± 5.18*** | 45.72 ± 3.72*** | 57.74 ± 5.19*** | 45.72 ± 3.73*** | 57.78 ± 5.18*** | 45.75 ± 3.71*** | 57.82 ± 5.20*** | 45.79 ± 3.74*** | 57.87 ± 5.23*** | 45.84 ± 3.76*** |
| CatBoostRegressor | 23.39 ± 3.68 * | 16.64 ± 2.55 ** | 22.80 ± 3.61 * | 16.31 ± 2.52 ** | 22.45 ± 3.50 * | 16.10 ± 2.48 ** | 22.27 ± 3.51 * | 15.98 ± 2.49 ** | 22.07 ± 3.44 * | 15.89 ± 2.44 ** | 21.85 ± 3.48 * | 15.72 ± 2.43 ** |
| Lightgbm | 21.09 ± 3.49 * | 14.74 ± 2.35 * | 19.28 ± 3.22 * | 13.57 ± 2.18 ** | 18.41 ± 3.06 * | 12.97 ± 2.11 * | 17.76 ± 3.08 * | 12.50 ± 2.10 * | 17.20 ± 2.94 * | 12.20 ± 2.06 * | 16.83 ± 3.01 * | 11.87 ± 2.03 * |
| ANN | 33.81 ± 4.96 ** | 26.15 ± 4.18 ** | 37.37 ± 3.66*** | 29.52 ± 2.79*** | 42.30 ± 5.26 ** | 34.32 ± 4.85 ** | 42.96 ± 3.71*** | 34.78 ± 3.47*** | 47.48 ± 5.10*** | 39.00 ± 4.18*** | 49.24 ± 4.50*** | 40.81 ± 3.67*** |
| XGBoost | 20.82 ± 3.43 * | 12.89 ± 5.72 * | 19.11 ± 3.03 * | 13.40 ± 1.98 ** | 18.28 ± 2.85 * | 12.80 ± 1.95 ** | 17.73 ± 2.77 * | 12.37 ± 1.83** | 17.24 ± 2.69 * | 12.12 ± 1.84 * | 16.82 ± 2.73 * | 11.83 ± 1.81 * |
| RandomForest | 19.65 ± 3.37 * | 12.94 ± 2.08 * | 17.96 ± 3.18 * | 11.55 ± 1.89 * | 17.21 ± 2.92 * | 10.92 ± 1.77 * | 16.75 ± 2.89 * | 10.52 ± 1.74 * | 16.21 ± 2.76 * | 10.17 ± 1.68 * | 15.95 ± 2.84 * | 9.91 ± 1.66 * |
| **KNN with best hyperparameter (**K=2**,** Weights='distance'**,** Metric='manhattan') | 8.27 ± 1.66 | 3.38 ± 0.48 | 5.65 ± 1.23 | **2.47 ± 0.34** | **5.56 ± 1.14** | 2.49 ± 0.34 | 5.61 ± 1.13 | 2.57 ± 0.33 | 5.59 ± 1.01 | 2.59 ± 0.34 | 5.60 ± 1.14 | 2.64 ± 0.35 |

*p ≤ 0.0001 **p ≤ 0.00001 ***p ≤ 0.000001.

## 4.2 Discussion

Table 2 and Table 3 show the obtained results, which are the Mean with standard deviation (Mean ± Std) of RMSE, MAE for datasets OhioT1DM V2018 and OhioT1DM V2020 in different windows sizes. Moreover, we apply 5-Fold Cross-Validation to assess the predictive performance of our models. Table 2 and Table 3 display the results of RMSE, MAE achieved through various methods for predicting blood glucose Levels over 30-Minute Timeframe. Furthermore, paired t-tests were conducted to ascertain the statistical significance of the KNN algorithm when employing its optimal hyperparameters.

The presented data in Table 2 and Table 3 indicate that the KNN regression algorithm with the optimal hyperparameter values of k=2, Weights='distance', and Metric='manhattan' is significant for predicting blood glucose levels over a 30-minutes timeframe. The KNN model achieved a minimum RMSE of 5.09 ± 0.91 mg/dl using a 24-hour window size, and an MAE of 2.42 ± 0.34 mg/dl with a 12-hour window size in OhioT1DM V2018 dataset. For the OhioT1DM V2020 dataset, the best results were an RMSE of 5.56 ± 1.14 mg/dl with a 12-hour window size, and an MAE of 2.47 ± 0.34 mg/dl achieved using an 8-hour window size.

Decision tree-based models, including RandomForest, XGBoost, Lightgbm, and CatBoostRegressor, showed that window sizes had a notable impact on RMSE and MAE. Specifically, the RMSE and MAE at a 24-hour window size were lower than those at a 4-hour window size.

The RandomForest model exhibited the best performance, yielding an RMSE of 15.65 ± 2.69 mg/dl, MAE of 9.50 ± 1.65 mg/dl with a 24-h window size in dataset OhioT1DM V2018 and an RMSE of 15.95 ± 2.84 mg/dl, MAE of 9.91 ± 1.66 mg/dl with a 24-h window size in dataset OhioT1DM V2020.

The linear regression (LR) model yielded highly similar results, and the selection of time window size did not significantly impact the performance, the RMSE values ranged from 23.63 ± 3.89 mg/dl to 23.86 ± 3.82 mg/dl in dataset OhioT1DM V2018 with the lowest RMSE of 23.63 ± 3.89 mg/dl observed when using a 4-h time window, and ranged from 24.64 ± 3.96 mg/dl to 24.92 ± 4.00 mg/dl in dataset OhioT1DM V2020 with the lowest RMSE of 24.64 ± 3.96 mg/dl observed when using a 4-h time window.

The performance of the SVR and ANN models was unsatisfactory, even with adjustments to the window size. The SVR model yielded RMSE values ranging from 58.51 ± 7.89 mg/dl to 58.66 ± 7.82 mg/dl in dataset OhioT1DM V2018, and ranging from 57.56 ± 5.06 mg/dl to 57.87 ± 5.23 mg/dl in dataset OhioT1DM V2020. Similarly, the ANN model exhibited RMSE values between 33.36 ± 4.13 mg/dl to 49.25 ± 3.85 mg/dl in dataset OhioT1DM V2018, and RMSE values between 33.81 ± 4.96 mg/dl to 49.24 ± 4.50 mg/dl in dataset OhioT1DM V2020.

As demonstrated in Table 4, the prediction of BGLs in diabetic patients in this study is comparable to the recent research reported in the literature using alternative methodologies. T. Hamdi et al. [5] explored the use of support vector regression and differential evolution techniques, they achieved a minimum RMSE of 10.78 mg/dl. T. Zhu et al. [6] developed a deep learning model that employed a dilated recurrent neural network architecture (DRNN), they achieved a minimum RMSE of 18.90 mg/dl. Dudukcu. H.V et al. [7] explored various sophisticated neural network designs, such as LSTM, WaveNet, and Gated Recurrent Units, they achieved a minimum RMSE of 21.90 mg/dl. T. Yang et al. [8] developed an autonomous deep learning model for personalized prediction of multiple blood glucose

measures, they achieved a minimum RMSE of 18.93 mg/dl. Shuvo. M. et al. [9] outlined a neural network architecture with shared and clustered hidden layers, they achieved a minimum RMSE of 16.06 ± 2.74 mg/dl.

Although the KNN algorithm is a traditional machine learning method, our study found it outperformed deep learning techniques, particularly with smaller datasets. KNN's performance, however, is sensitive to the choice of the value for k. To optimize the algorithm's effectiveness, we evaluated various k values, weights, distance metrics, and window sizes. To the best of our knowledge, no prior research has compared KNN against modern methods while systematically testing these parameters.

Our study yielded promising results. For the OhioT1DM V2018 dataset, we achieved a minimum RMSE of 5.09 ± 0.91 mg/dl using a 24-hour window size, and an MAE of 2.42 ± 0.34 mg/dl with a 12-hour window size. On the OhioT1DM V2020 dataset, the best results were an RMSE of 5.56 ± 1.14 mg/dl (12-hour window size) and an MAE of 2.47 ± 0.34 mg/dl (8-hour window size), obtained by employing a k-nearest neighbors algorithm with the following parameter settings: (k = 2, weights = 'distance', and metric = 'manhattan').

Table 4: Comparison of our approaches with prior approaches in the literature

| **Authors** | **Techniques** | **Datasets** | **Forecast Horizon** | **RMSE (mg/dL)** | **MAE (mg/dL)** |
|---|---|---|---|---|---|
| T. Hamdi et al. [5] | SVR based on DE algorithm | 12 type1 diabetes | 30_min | 10.78 | - |
| T. Zhu et al. [6] | DRNN | OhioT1DM | 30_min | 18.90 | - |
| Dudukcu. H.V et al. [7] | LSTM, Wave-Net, GRU | OhioT1DM | 30_min | 21.90 | - |
| T. Yang et al. [8] | AC-DLF | OhioT1DM | 30_min | 18.93 | - |
| Shuvo. M. et al. [9] | D-MTL | OhioT1DM | 30_min | 16.06 ± 2.74 | 10.64 ± 1.35 |
| **Our Methods** | **KNN With the best hyperparameter** | **OhioT1DM V2018** | 30_min | **5.09 ± 0.91** | **2.42 ± 0.34** |
| | | **OhioT1DM V2020** | 30_min | **5.56 ± 1.14** | **2.47 ± 0.34** |

# 5 Limitations

This study encountered certain limitations, which we address first. The K-Nearest Neighbors algorithm, while capable of delivering strong predictive performance, particularly when dealing with datasets exhibiting distinct separation boundaries, incurs significant computational costs during the prediction phase. The algorithm's need to compare the query instance against every point within the training dataset to identify the closest neighbors leads to a time complexity of O (n × d), where 'n' represents the count of training samples and 'd' denotes the dimensionality. In embedded or real-time systems, where low latency and energy efficiency are paramount, the aforementioned factor can represent a substantial limitation. Consequently, a balance must be struck between predictive accuracy and execution-time performance, necessitating careful evaluation before implementing KNN in these operational contexts. To mitigate this problem, techniques such as approximate nearest neighbor search, dimensionality reduction, or prototype selection can be employed, representing the future direction of our experiments.

A further limitation is that models trained exclusively on continuous glucose monitoring data might overfit to individual patient-specific patterns, hindering their generalizability across diverse patient populations or in response to altered daily routines. Furthermore, CGM-based models primarily demonstrate efficacy in short-term glucose trend predictions, with a constrained capacity to forecast long-term trends without integrating other factors such as bolus doses, basal rates, interim basal rates, meal consumption, exercise routines, and basal heart rate.

# 6 Conclusion

Accurately predicting BGLs in diabetes is crucial, as this will enable the artificial pancreas to secrete the necessary amount of insulin. This paper presents a functioning method for forecasting BGLs in real individuals with type 1 diabetes, using data from continuous glucose monitoring. The results indicate that the k-nearest neighbors' algorithm with optimal hyperparameters (k=2, Weight='distance', Metric='manhattan') was effective in predicting blood glucose levels over the short-term (30-minute forecast horizon) in all sliding windows size 8-hours, 12-hours, 16-hours, 20-hours, 24-hours. Additionally, the window size had an effect on reducing the error rate of the predictions. Specifically, the RMSE and MAE at 8-hours, 12-hours, 16-hours, 20-hours, 24-hours window size were lower than those at a 4-hour window size.

The model has been evaluated on the two versions of OhioT1DM V2018 and V2020. These datasets contain real values of measurements taken from real diabetic patients living in their natural environments, which considered as strong point to validate our findings. In future research, we suggest to integrate and experiment this model into CGM device and clinical decision-making tools.

# Acknowledgment

This study was supported by the General Directorate for Scientific Research and Technological Development (DGRSDT), Algeria. Moreover, it was conducted as part of the research activities at LIMED laboratory, which is affiliated with the Exact Sciences Faculty at Bejaia University.

# References

[1] Liu, K., Li, L., Ma, Y., Jiang, J., Liu, Z., Ye, Z., … Yi, W. "Machine Learning Models for Blood Glucose Level Prediction in Patients With Diabetes Mellitus: Systematic Review and

Network Meta-Analysis". *JMIR medical informatics*, *11*(1). 2023. DOI:10.2196/47833

[2] Prendin, F., Del Favero, S., Vettoretti, M., Sparacino, G., Facchinetti, A. "Forecasting of Glucose Levels and Hypoglycemic Events: Head-to-Head Comparison of Linear and Nonlinear Data-Driven Algorithms Based on Continuous Glucose Monitoring Data Only". *Sensors 2021, Vol. 21, Page 1647*, *21*(5), pp. 1647. 2021. DOI:10.3390/S21051647

[3] D'Antoni, F., Merone, M., Piemonte, V., Iannello, G., Soda, P. "Auto-Regressive Time Delayed jump neural network for blood glucose levels forecasting". *Knowledge-Based Systems*, *203*, pp. 106134. 2020. DOI:10.1016/J.KNOSYS.2020.106134

[4] Daskalaki, E., Prountzou, A., Diem, P., Mougiakakou, S. G. "Real-Time Adaptive Models for the Personalized Prediction of Glycemic Profile in Type 1 Diabetes Patients". *https://home.liebertpub.com/dia*, *14*(2), pp. 168–174. 2012. DOI:10.1089/DIA.2011.0093

[5] Hamdi, T., Ben Ali, J., Di Costanzo, V., Fnaiech, F., Moreau, E., Ginoux, J. M. "Accurate prediction of continuous blood glucose based on support vector regression and differential evolution algorithm". *Biocybernetics and Biomedical Engineering*, *38*(2), pp. 362–372. 2018. DOI:10.1016/J.BBE.2018.02.005

[6] Zhu, T., Li, K., Chen, J., Herrero, P., Georgiou, P. "Dilated Recurrent Neural Networks for Glucose Forecasting in Type 1 Diabetes". *Journal of Healthcare Informatics Research*, *4*(3), pp. 308–324. 2020. DOI:10.1007/S41666-020-00068-2/FIGURES/7

[7] Dudukcu, H. V., Taskiran, M., Yildirim, T. "Blood glucose prediction with deep neural networks using weighted decision level fusion". *Biocybernetics and Biomedical Engineering*, *41*(3), pp. 1208–1223. 2021. DOI:10.1016/J.BBE.2021.08.007

[8] Yang, T., Yu, X., Ma, N., Wu, R., Li, H. "An autonomous channel deep learning framework for blood glucose prediction". *Applied Soft Computing*, *120*, pp. 108636. 2022. DOI:10.1016/J.ASOC.2022.108636

[9] Shuvo, M. M. H., Islam, S. K. "Deep Multitask Learning by Stacked Long Short-Term Memory for Predicting Personalized Blood Glucose Concentration". *IEEE journal of biomedical and health informatics*, *PP*(3), pp. 1612–1623. 2023. DOI:10.1109/JBHI.2022.3233486

[10] Marling, C., Bunescu, R. "The OhioT1DM Dataset for Blood Glucose Level Prediction: Update 2020". *CEUR workshop proceedings*, *2675*, pp. 71. 2020. Retrieved from /pmc/articles/PMC7881904/ PMID: 33584164; PMCID: PMC7881904.

[11] Cao, N., Yan, X. E., Zhang, L., Xu, G., Ma, J. "Hybrid K-Nearest Neighbors Models with Metaheuristic Optimization for Predicting Undrained Shear Strength". *Informatica*, *49*(25), pp. 125–144. 2025. DOI:10.31449/INF.V49I25.7723

[12] Xiong, Y. "Development of an AI-Driven Model for Drug Sales Prediction Using Enhanced Golden Eagle Optimization and XGBoost Algorithm". *Informatica*, *49*(17), pp. 37–50. 2025. DOI:10.31449/INF.V49I17.7491

## Abbreviation

**ANN:** Artificial neural network; **BG**: Blood Glucose; **BGLs**: Blood Glucose levels; **CatBoostClassifier:** Categorical Boosting Classifier; **CGM:** Continuous Glucose Monitor; **KNN:** K-nearest neighbor; **LightGBM:** Light gradient-boosting machine; **RF:** Random Forest; **SVM:** Support Vector Machine; **T1DM:** Type 1 Diabetes mellitus; **XGBoost:** Extreme Gradient Boosting; **RMSE:** Root Mean Square Error; **h:** hours; **MAE:** Mean Absolutely Error.