
الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique



المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique



Département: Génie Industriel
Université hôte: Université TÉLUQ

*End-of-study project dissertation for obtaining the State Engineer's degree in Industrial
Engineering
Specialization: Data Science & Artificial Intelligence*

Prediction of Hypoglycemia Episodes in Type 1 Diabetes Patients

Presented by: Mohamed Merouane LAKEHAL

Under the direction of **Mrs. Samia BELDJOU DI**, ENP
& **Mrs. Khadidja HENNI**, TÉLUQ
& **Mrs. Leila HAMDAD**, ESI

Presented and defended publicly on **(25/06/2025)**

Jury Composition:

President	Mr. Iskander ZOUAGHI	MCA	ENP
Examiner	Mrs. Imene BARECHE	MAB	ENP
Promoter	Mrs. Samia BELDJOU DI	MCA	ENP

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique



المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique



Département: Génie Industriel
Université hôte: Université TÉLUQ

*End-of-study project dissertation for obtaining the State Engineer's degree in Industrial
Engineering
Specialization: Data Science & Artificial Intelligence*

Prediction of Hypoglycemia Episodes in Type 1 Diabetes Patients

Presented by: Mohamed Merouane LAKEHAL

Under the direction of **Mrs. Samia BELDJOU DI**, ENP
& **Mrs. Khadidja HENNI**, TÉLUQ
& **Mrs. Leila HAMDAD**, ESI

Presented and defended publicly on **(25/06/2025)**

Jury Composition:

President	Mr. Iskander ZOUAGHI	MCA	ENP
Examiner	Mrs. Imene BARECHE	MAB	ENP
Promoter	Mrs. Samia BELDJOU DI	MCA	ENP

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique



المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique



Département: Génie Industriel
Université hôte: Université TÉLUQ

*Mémoire de Projet de fin d'Etudes pour l'obtention du Diplôme d'Ingénieur d'Etat en Génie
Industriel
Specialization: Data Science & Intelligence Artificielle*

Prédiction des épisodes d'hypoglycémie chez les patients atteints de diabète de type 1

Présenté par : **Mohamed Merouane LAKEHAL**

Sous la Direction de **Mme. Samia BELDJOUDI**, ENP
& **Mme. Khadidja HENNI**, TÉLUQ
& **Mme. Leila HAMDAD**, ESI

Présenté et soutenue publiquement le **(25/06/2025)**

Jury Composition:

Président	M. Iskander ZOUAGHI	MCA	ENP
Examineur	Mme. Imene BARECHE	MAB	ENP
Promoteur	Mme. Samia BELDJOUDI	MCA	ENP

الملخص

الهبوط السكري، الذي يُعرف بمستوى جلوكوز في الدم أقل من ٠.٧ ملغم/ديسيلتر، يمثل خطراً كبيراً للأفراد المصابين بمرض السكري من النوع الأول، وقد يؤدي إلى نتائج خطيرة مثل التشنجات أو فقدان الوعي إذا لم يُعالج في الوقت المناسب.

تُعاني أنظمة المراقبة المستمرة للجلوكوز (CGM)، رغم كونها أقل تدخلاً من اختبار وخز الأصبع، من تأخر فسيولوجي تتراوح مدته بين ٥ و ٢٠ دقيقة بين مستويات الجلوكوز في الدم والنسيج الخلالي، مما يحد من قدرتها على إنذار المرضى مبكراً.

تقترح هذه الدراسة نموذجاً تنبؤياً لتوقع نوبات انخفاض السكر في وقت مبكر، مما يساعد المرضى على اتخاذ إجراءات وقائية مثل تناول الكربوهيدرات.

باستخدام بيانات CGM التسلسلية، يستكشف النموذج مدخلات أحادية المتغير فقط CGM ومتعددة المتغيرات (بما في ذلك الإنسولين وتناول الكربوهيدرات). كما يعالج تحدي عدم توازن البيانات، مع التركيز على تحقيق دقة واسترجاع عاليين لتقليل الإنذارات الكاذبة.

تُظهر النتائج أن النماذج أحادية المتغير تؤدي أداءً مماثلاً للنماذج متعددة المتغيرات، مما يجعلها عملية للاستخدام في الواقع. كما أن النماذج القائمة على الانحدار تتمتع بقدرة تعميم أفضل عبر ظروف الاختبار مقارنة بنماذج التصنيف.

يُعزز التحليل بشبكة خطأ كلارك الصلاحية السريعة للنموذج، حيث تقع أكثر من ٨٩٪ من التنبؤات في المناطق الآمنة A و B. يدعم هذا النهج إدارة السكري بشكل أكثر أماناً واستباقية من خلال التنبؤ المبكر بهبوط السكر بالاعتماد على (CGM).

الكلمات المفتاحية: تنبؤ بنقص السكر في الدم، المراقبة المستمرة للجلوكوز (CGM)، تنبؤ بالسلاسل الزمنية، اختلال توازن البيانات، إدارة مرض السكري، نظام إنذار مبكر، التعلم الآلي،

Résumé

L'hypoglycémie, définie comme un taux de glucose sanguin inférieur à 70 mg/dL, constitue un risque majeur pour les personnes atteintes de diabète de type 1 (DT1), pouvant entraîner des complications graves telles que des convulsions ou une perte de conscience si elle n'est pas prise en charge à temps. Les systèmes de surveillance continue du glucose (CGM), bien qu'ils soient moins invasifs que les tests au doigt, présentent un décalage physiologique de 5 à 20 minutes entre les niveaux de glucose sanguin et interstitiel, ce qui limite leur capacité à alerter précocement les patients.

Cette étude propose un modèle prédictif permettant d'anticiper les épisodes hypoglycémiques, afin d'aider les patients à prendre des mesures préventives comme l'ingestion de glucides. En utilisant des données CGM en séries temporelles, le modèle explore des entrées univariées (CGM seul) et multivariées (incluant l'insuline et l'apport en glucides). Il aborde également le problème du déséquilibre des données, avec un accent mis sur la précision et le rappel pour limiter les fausses alertes.

Les résultats montrent que les modèles univariés offrent des performances comparables aux modèles multivariés, ce qui les rend pratiques pour une utilisation réelle. Les modèles de régression généralisent également mieux que les modèles de classification. La validité clinique du modèle est confirmée par l'analyse de la grille d'erreur de Clarke, avec plus de 98 % des prédictions situées dans les zones sûres (A et B).

Cette approche favorise une gestion du diabète plus sûre et proactive grâce à la prédiction précoce de l'hypoglycémie basée sur les CGM.

Keywords : Prédiction de l'hypoglycémie, Surveillance continue du glucose (CGM), Prévision en séries temporelles, Déséquilibre des données, Gestion du diabète, Système d'alerte précoce, Apprentissage automatique,

Abstract

Hypoglycemia, defined as a blood glucose level below 70 mg/dL, is a serious risk for individuals with Type 1 Diabetes Mellitus (T1DM), potentially causing severe outcomes such as seizures or unconsciousness if not addressed in time. Continuous Glucose Monitoring (CGM) systems, though less invasive than finger-prick testing, suffer from a physiological lag of 5–20 minutes between blood and interstitial glucose levels, limiting their ability to warn patients early.

This study proposes a predictive model to anticipate hypoglycemic events ahead of time, helping patients take preventive actions like carbohydrate intake. Using time-series CGM data, the model explores both univariate (CGM-only) and multivariate (including insulin and carbohydrate intake) inputs. It also addresses the challenge of data imbalance, with a focus on achieving high precision and recall to reduce false alarms.

Results show that univariate models perform comparably to multivariate ones, making them practical for real-world use. Regression-based models also generalize better across test conditions than classification models. The model's clinical validity is supported by Clarke error grid analysis, where over 98% of predictions fall in safe zones (A and B).

This approach supports safer, proactive diabetes management through timely, CGM-based hypoglycemia prediction.

Keywords: Hypoglycemia prediction, Continuous glucose monitoring (CGM), Time series forecasting, Data imbalance, Diabetes management, Early warning system, Machine learning,

Acknowledgments

First and foremost, I thank God Almighty for granting me the strength, patience, and guidance to complete this work.

I am deeply grateful to my parents, whose unconditional love, constant encouragement, and sacrifices have been the foundation of my journey. I also extend my heartfelt thanks to my young sister and my entire family for their unwavering support, understanding, and belief in me.

I would like to express my sincere appreciation to my supervisors, Dr. BELDJOUDI, Dr. HENNI and Dr. HAMDAD, for their guidance, constructive feedback, and encouragement throughout the course of this research.

I would like also to express my sincere gratitude to the faculty and staff of the Industrial Engineering Department for their unwavering support and dedication throughout the past three years of academic formation.

Special thanks to the faculty and staff of Université TÉLUQ for providing a supportive academic environment and access to the resources necessary for this project.

Finally, I am thankful to my colleagues and friends for their support, collaboration, and shared moments of motivation and challenge during this academic endeavor.

Contents

List of Tables

List of Figures

List of Algorithms

List of Abbreviations

Introduction	14
1 Overview of the AI Field	18
1.1 Definition of Artificial Intelligence	18
1.2 Machine Learning	19
1.3 Deep Learning	20
1.3.1 Challenges in Optimization	22
1.3.2 Loss Functions	22
1.4 Deep Learning Architectures	23
1.5 Time Series	27
1.6 Explainable AI (XAI)	28
2 AI for HealthCare	30
2.1 Applications of AI in Healthcare	30
2.2 Trustworthy AI in Healthcare	31
2.3 Related Work	31
2.4 Problem Overview	35
3 Methodology	37
3.1 Proposed pipeline	37
3.2 Preprocessing	37
3.2.1 Data Augmentation	39

3.3	Diffusion Models for data augmentation	40
3.3.1	Network Architecture	42
3.3.2	Mathematical Formulation	42
3.3.3	Implementation Details	44
3.3.4	Training Procedure	44
3.3.5	Advantages for 1D Data	45
3.4	Hyperparameter Tuning	45
3.4.1	Genetic Algorithm for Hyperparameter Optimization	46
3.5	Models	47
3.5.1	Random Forest	47
3.5.2	XGBoost	48
3.5.3	Long Short-Term Memory (LSTM)	48
3.5.4	Convolutional Neural Network (CNN)	50
3.5.5	Inception-LSTM	51
3.5.6	Transformer	52
3.5.7	S4 (Structured State Space Models)	54
3.5.8	Patching Technique	58
3.6	Design Choices	59
3.7	Training Set-Up	60
3.7.1	Half-Precision Floating-Point Training in Deep Neural Networks	62
3.8	Conclusion	63
4	Evaluation, Results & Discussions	64
4.1	Datasets	64
4.2	Metrics of Evaluation	68
4.3	Evaluation	69
4.3.1	Multivariate	69
4.3.2	Univariate	73
4.3.3	Data Augmentation	80
4.3.4	Model Explainability	82
4.3.5	Out of distribution evaluation:	83
4.3.6	Time Gain	87
4.4	Discussions	88
	Conclusion	94
	Bibliographie	96

List of Tables

Table 2.1	Summary of Related Works on Glycemic Event Prediction . . .	35
Table 3.1	Components of U-net.	42
Table 4.1	Comparison of datasets used in the study.	67
Table 4.2	Binary Classification evaluation for Different Prediction Horizons: Multivariate approach.	71
Table 4.3	Three-Class Classification evaluation for Different Prediction Horizons: Multivariate approach.	72
Table 4.4	Regression performance metrics and hypoglycemia F1-scores across prediction horizons (10, 20, 30 mins): Multivariate approach.	73
Table 4.5	Binary Classification evaluation (Hypo, Eu, Hype) for Different Prediction Horizons: Univariate approach.	74
Table 4.6	Three-Class Classification Metrics (Hypo, Eu, Hype) for Different Prediction Horizons: Univariate approach.	76
Table 4.7	Regression Performance for Glucose Prediction (Autoregressive and Multi-Horizon Settings).	78
Table 4.8	Evaluation of classification performance in the case of data augmentation.	81
Table 4.9	Evaluation of regression performance in the case of data augmentation.	81
Table 4.10	Out-of-distribution Evaluation for Classification Task (F1-score)	83
Table 4.11	Regression Out-of-Distribution evaluation (RMSE)(mg/dl) . .	84
Table 4.12	Out-of-Distribution evaluation of classification performance based on series forecasting	85
Table 4.13	Model performance comparison across different train-test combinations	88
Table 4.14	Clinical evaluation of trained models	92

List of Figures

Figure 1	Continuous Glucose Monitoring device	15
Figure 1.1	Different types of AI	20
Figure 1.2	Perceptron	21
Figure 1.3	Gradient Descent algorithm	21
Figure 1.4	Convolutional neural networks for signals	24
Figure 1.5	Recurrent Neural Networks	24
Figure 1.6	Attention mechanism	26
Figure 1.7	State Space Models	27
Figure 3.1	Proposed pipeline.	38
Figure 3.2	Example of a CGM signal.	39
Figure 3.3	Forward and reverse process in DDPM	40
Figure 3.4	U-Net architecture for 1D diffusion models with encoder-decoder structure and skip connections.	43
Figure 3.5	Long Short Term Memory cell.	48
Figure 3.6	Inception Block.	51
Figure 3.7	GPT-1 like model for time series.	53
Figure 3.8	Different Computational Forms of S4.	56
Figure 3.9	Patching technique in the case of Transformer	59
Figure 4.1	Example of synthetic CGM signal.	80
Figure 4.2	Explanation of trained models using SHAP.	82
Figure 4.3	Explanation of trained models using SHAP.	82
Figure 4.4	Histogram of forecasting delays.	90
Figure 4.5	Clarke Error Grid.	90
Figure 4.6	Clarke Error Grid for transformer.	91

List of Algorithms

1	Training U-Net for 1D Diffusion	45
2	Genetic Algorithm for Hyperparameter Optimization	47
3	Model Training with Optional Mixed Precision and Early Stopping	61

List of Abbreviations

- **DM** : Diabetes Mellitus
- **T1DM** : Type 1 Diabetes Mellitus
- **CGM** : Continuous Glucose Monitoring device
- **IDDM** : Insulin-Dependent Diabetes Mellitus
- **BG** : Blood Glucose
- **CHO** : Carbohydrates
- **BGS** : Blood Glucose Self-monitoring
- **MARD** : Mean Absolute Relative Difference
- **AI** : Artificial Intelligence
- **ML** : Machine Learning
- **DL** : Deep Learning
- **NN** : Neural Network
- **TLU** : Threshold Logic Unit
- **CNN** : Convolutional Neural Network
- **RNN** : Recurrent Neural Network
- **BPTT** : Backpropagation Through Time
- **SSM** : State Space Models
- **SHAP** : SHapley Additive exPlanations

- **TRF** : Transformer
- **DDPM** : Denoising Diffusion Probabilistic Models
- **RF** : Random Forest
- **XGB** : eXtreme Gradient Boosting

Introduction

Diabetes mellitus (DM) is a metabolic disorder characterized by chronically elevated blood glucose levels, also known as hyperglycemia. This disorder results from anomalies in insulin secretion, insulin action, or both.

Hyperglycemia and its associated dysfunctions in carbohydrate, fat, and protein metabolism affect multiple organs in the human body and disrupt their normal functioning. These anomalies progress gradually and are exacerbated mainly by adverse glycemic events. By causing complications and impairing the microvascular and macrovascular systems, DM leads to organ damage, dysfunction, and, ultimately, failure. Based on the etiology and pathogenesis of DM, four main types can be distinguished: type 1 diabetes mellitus (T1DM), type 2 diabetes mellitus (T2DM), gestational diabetes mellitus (GDM), and diabetes caused by specific conditions or disorders[1].

Type 1 diabetes mellitus (T1DM), previously known as insulin-dependent diabetes mellitus (IDDM), constitutes about 5–10% of diabetes cases worldwide. It is an autoimmune disorder characterized by the destruction of pancreatic β -cells responsible for insulin secretion, ultimately leading to hyperglycemia. T1DM patients require continuous management of their blood glucose (BG) levels using exogenous insulin to maintain glucose concentrations within the euglycemic range: 70 – 180 mg/dl. Incorrect estimation of the required amount of exogenous insulin can sometimes lead to hypoglycemia, defined as BG levels below the normal range (< 70 mg/dl). This dangerous condition can lead to coma or even death if not treated swiftly.

The traditional treatment for hypoglycemic episodes involves taking 16 g of carbohydrates (CHO) every 15 min until the event is resolved. Recent studies [2] suggest a preventive approach, to administer glucose at higher BG levels (80 or 90 mg/dl). This has shown to reduce both the incidence and severity of hypoglycemic events. However, this requires continuous monitoring of blood glucose levels, which is impractical with a

glucometer due to the need for multiple daily finger pricks, which can be inconvenient or painful.



Figure 1: Continuous Glucose Monitoring device

To address this challenge, a device known as Continuous Glucose Monitoring (CGM) [Figure 1](#) has been introduced for T1DM patients. It provides a continuous stream of glucose readings (one measurement every 5 min) from interstitial fluid, enabling a more comprehensive understanding of glucose trends and fluctuations. More importantly, CGM offers real-time glucose monitoring, facilitating Blood Glucose Self-monitoring (BGS). Since CGM measures interstitial rather than capillary glucose, its readings tend to lag behind blood glucose levels by up to 15 min depending on the device generation and cartridge age (performance decreases over time, with cartridges usable for up to seven consecutive days). Performance of modern CGM devices tend to be described using the MARD (mean absolute relative difference) [\[3\]](#), which is the mean difference, expressed in absolute terms, between the reference (often Blood Glucose Monitoring device "glucometer") and interstitial glucose divided by the reference. Modern CGM devices have a MARD of about 8%, an improvement that reaches 1% compared to previous generation.

In this project, we try to build a reliable and resilient hypoglycemia prediction model based on modern time series techniques and deep learning architectures, in order to help patients take preventive measurements in the right time before a critical situation arises.

About this Thesis

This thesis investigates the application of intelligent systems in improving the management of Type 1 Diabetes Mellitus (T1DM) through real-time analysis of Continuous Glucose Monitoring (CGM) data. The work is motivated by the clinical challenge of maintaining blood glucose levels within a safe range, particularly the need to prevent acute hypoglycemia events caused by inaccurate insulin dosing or delayed treatment.

The core objective of this research is to explore how CGM data, which offers granular and continuous insight into glucose trends, can be used to develop predictive models or decision-support tools that enhance glycemic control. By leveraging modern computational techniques—such as time series analysis, machine learning, or rule-based alert systems—this thesis aims to demonstrate how technology can provide more proactive, personalized, and safer glucose management solutions for individuals with T1DM.

The work includes a review of existing CGM technology and glucose monitoring strategies, followed by the design and implementation of a prototype system capable of analyzing CGM data in real-time. The thesis also evaluates the effectiveness of the proposed system in identifying or predicting adverse glycemic events, and reflects on the practical implications, limitations, and future directions for smarter diabetes care.

This research contributes to the broader goal of digital health innovation by integrating biomedical data with intelligent computation, ultimately empowering patients and clinicians to make informed and timely decisions in managing chronic conditions like diabetes.

Chapter List

This section provides a list of all the chapters in the thesis along with a brief summary of each:

Chapter 1 – *Background Research*. This chapter deals with the foundational concepts of Artificial Intelligence (AI), covering its definitions, core subfields—including Machine Learning and Deep Learning—and specialized areas such as time series analysis and model explainability.

Chapter 2 – *Design*. This chapter deals with the uses and standards of AI in Medical sector, focusing on vital signs monitoring and specially CGM signals. It also provides

an overview of previous works related to our project.

Chapter 3 – *Methodologies*. This chapter deals with the different stages and elements of the proposed solution. Delving deep into the technical knowledge and the decisions taking during the design.

Chapter 4 – *Evaluation & Testing*. This chapter presents the evaluation of the implemented solution, and the observations that follows the results.

Conclusion

In this chapter, the foundational background necessary to understand the broader context of this thesis has been established. Diabetes mellitus, particularly Type 1, has been presented as a complex and demanding condition that necessitates vigilant glucose regulation to avoid life-threatening complications such as hypoglycemia. Traditional monitoring techniques, while effective to a degree, pose significant limitations in terms of practicality and real-time responsiveness. The emergence of Continuous Glucose Monitoring (CGM) systems addresses many of these limitations by providing patients and clinicians with continuous, actionable data. This capability not only improves immediate glycemic control but also opens the door to predictive modeling and proactive interventions. The subsequent chapters will delve into the design, implementation, and evaluation of systems that leverage CGM data to support more intelligent and responsive diabetes management strategies.

Chapter 1

Overview of the AI Field

Artificial Intelligence (AI) has rapidly evolved into a cornerstone technology with wide-ranging applications across science, industry, and society. This chapter introduces foundational concepts in AI, focusing on its definitions, key subfields such as Machine Learning and Deep Learning, and specialized topics like time series analysis and explainability. The goal is to provide a comprehensive overview to ground the reader in the terminology and methodologies that underpin contemporary AI research and applications.

1.1 Definition of Artificial Intelligence

There is no universally accepted definition for AI [4], as it involves imitating complex human cognitive abilities that we do not yet fully understand. Broadly speaking, AI could be equated with algorithms — including operations as simple as those performed by a pocket calculator or even the instructions in a cookbook.

In a more precise sense, AI refers to the imitation of human intelligence. Under this interpretation, many current models would not qualify as they are far from reaching human-level intelligence.

A more practical and widely used definition describes AI as a technology enabling machines to simulate various complex human skills.

AI is a scientific discipline that seeks to understand intelligence through the design and construction of intelligent machines.

Based on algorithmic principles, we can categorize two main approaches:

- **Rule-based systems** (also known as symbolic AI), where logic is explicitly encoded using a series of IF-THEN statements.
- **Machine Learning (ML)**, which is the approach most relevant to our context, involving learning patterns directly from data.

1.2 Machine Learning

Machine Learning is the science and art of programming computers so they can learn from data [5]. A classic engineering-oriented definition, proposed by Tom Mitchell in 1977, states:

A computer program is said to learn from experience E with respect to some task T and some performance measure P , if its performance on T , as measured by P , improves with experience E .

ML systems can be classified according to the kind of supervision they receive during training:

- **Supervised Learning:** Training data contains desired outputs (labels). Example: image classification into cats and dogs.
- **Unsupervised Learning:** Training data contains no labels; the algorithm discovers hidden patterns.
- **Semi-supervised Learning:** Training data is partially labeled, useful when labeling is expensive.
- **Reinforcement Learning:** An agent learns to make decisions by interacting with an environment and receiving rewards or penalties.

Another classification criterion is the approach to generalization:

- **Instance-based:** Also called memory-based learning; stores training data and generalizes by comparing new cases to recorded examples.
- **Model-based:** Builds a model from training data, extracting patterns to make predictions.

Based on the previous definition, it's clear that our problem falls under **supervised**

learning, as we can produce labels for each sequence based on the upcoming measurements in the CGM sequence. We plan to focus more on **model-based** approaches, as it typically offer better performance and generalization.

1.3 Deep Learning

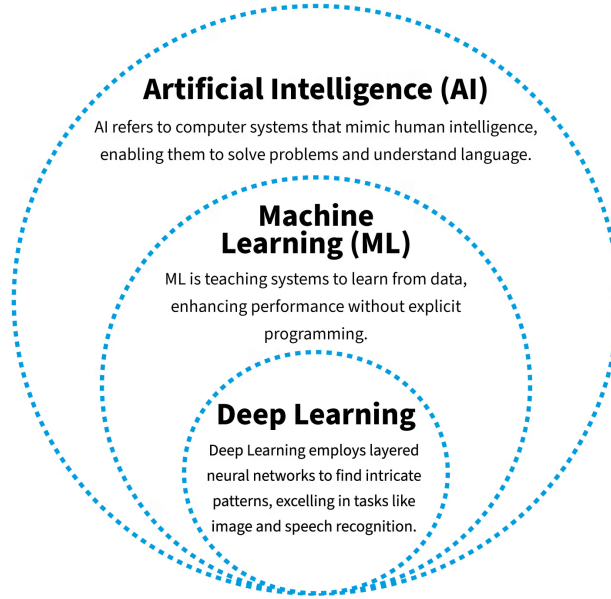


Figure 1.1: Different types of AI

Deep Learning is a subfield of Machine Learning [Figure 1.1](#) based on artificial neural networks with multiple processing layers extracting progressively higher-level features from data. Early inspiration came from biological neuroscience, but modern architectures do not attempt to replicate brain functions exactly.

The fundamental building block is the **perceptron** [Figure 1.2](#), invented by Frank Rosenblatt in 1957. It is based on a threshold logic unit (TLU), where inputs and outputs are numeric, each input has a weight, and the TLU computes a weighted sum followed by a step function such as the Heaviside function:

$$y = \begin{cases} 1 & \text{if } \sum_i w_i x_i + b > 0 \\ 0 & \text{otherwise} \end{cases} \quad (1.1)$$

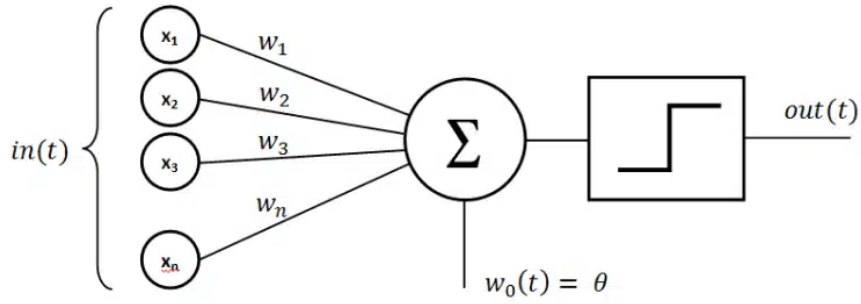


Figure 1.2: Perceptron

Here, x_i are inputs, w_i their weights, and b is a bias term.

A single TLU can be used to build a simple linear binary classifier. It computes the linear combination of the input, and outputs the positive class if the weighted sum is greater than zero; otherwise it will be the negative class.

Modern deep learning architectures combine many such units with diverse activation functions, such as ReLU, organized to exploit specific data properties (e.g., CNNs for spatial locality in images).

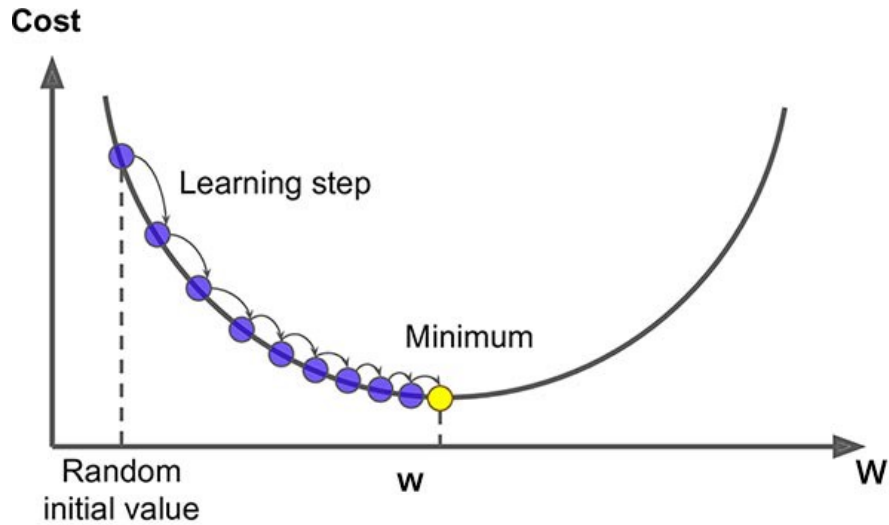


Figure 1.3: Gradient Descent algorithm

Neural networks are trained to minimize a **cost function** $J(\theta)$, where θ denotes the model parameters [Figure 1.3](#). Optimization is commonly performed using variants of

Gradient Descent , which iteratively updates parameters:

$$\theta \leftarrow \theta - \alpha \nabla_{\theta} J(\theta) \quad (1.2)$$

where α is the learning rate and $\nabla_{\theta} J(\theta)$ is the loss gradient with respect to the parameters. Training continue until the gradients vanish (local minimum) or a convergence criterion is met.

1.3.1 Challenges in Optimization

Training deep networks can encounter several issues:

- **Vanishing gradients:** Gradients become too small as they backpropagate, slowing learning. Techniques to mitigate include gradient clipping, careful initialization, ReLU activation, and residual connections.
- **Exploding gradients:** Excessively large gradients causing instability, mitigated by clipping or lowering the learning rate.
- **Overfitting:** Model learns noise rather than patterns, harming generalization. Mitigations include regularization and early stopping.
- **Underfitting:** Model too simple to capture patterns. Using more complex models or better architectures helps.
- **Hyperparameter tuning:** Critical for good performance (learning rate, batch size, etc.)

In practice, the classical gradient descent isn't used but its variants, to optimize model parameters more efficiently and effectively by adapting the basic algorithm to different situations. The most important ones are Mini-Batch Gradient Descent and Adaptive Moment Estimation (Adam) [6].

1.3.2 Loss Functions

The learning process involves optimizing the weights to minimize the loss. The loss is computed by the loss function (or objective function), which quantifies how far off the model's prediction is from the actual target. The choice of loss function depends on the nature of the problem:

- **Regression:** Mean Squared Error, Mean Absolute Error, Huber Loss.
- **Classification:** Cross-Entropy Loss, Negative Log-Likelihood.
- **Ranking:** Margin Ranking Loss.
- **Bayesian:** Kullback-Leibler Divergence.
- **Unsupervised:** Triplet Margin Loss, Hinge Embedding Loss.

1.4 Deep Learning Architectures

Deep neural networks learn to distinguish important patterns in the training data by using a series of non-linear layers. By passing the input into the layers, the model is trained to construct a latent representation $\mathbf{z}_t$ that stores the required information from the input for the decision.

In time series settings, the encoded information not only is extracted from the individual measurements at each time step, but also from the temporal order in which the sequence was provided. In other terms, historical dependencies are also taken into consideration.

$$f(y_{t-k:t}, \mathbf{x}_{t-k:t}, \mathbf{s}) = g_{\text{dec}}(\mathbf{z}_t), \quad (1.3)$$

and

$$\mathbf{z}_t = g_{\text{enc}}(y_{t-k:t}, \mathbf{x}_{t-k:t}, \mathbf{s}), \quad (1.4)$$

where $g_{\text{enc}}(\cdot)$ and $g_{\text{dec}}(\cdot)$ are encoder and decoder functions, respectively. These encoders and decoders act as the fundamental building block for deep learning architectures[7]. In the case of time series (and more generally in deep learning), the choice of the architecture will determine the types of relationships learned during training.

Convolutional Neural Networks (CNNs)

In time series, 1D convolutions [Figure 1.4](#) are used to extract local temporal patterns within a fixed-size window. For an input sequence $\mathbf{x} = [x_1, x_2, \dots, x_T]$, a 1D convolution applies a filter $\mathbf{w} = [w_1, w_2, \dots, w_k]$ to produce features:

$$(\mathbf{x} * \mathbf{w})_t = \sum_{i=0}^{k-1} w_i x_{t+i} \quad (1.5)$$

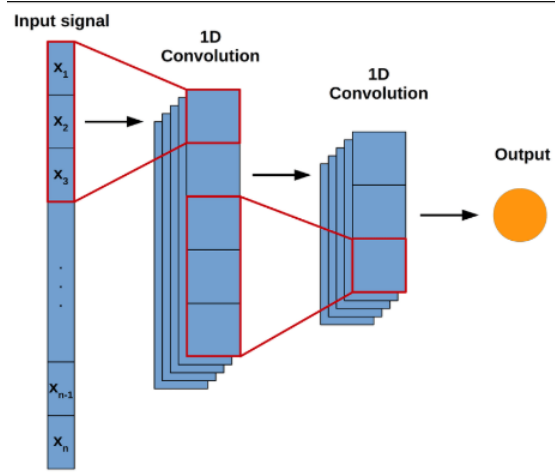


Figure 1.4: Convolutional neural networks for signals

where k is the kernel size and the operation slides over the temporal dimension capturing local dependencies.

Recurrent Neural Networks (RNNs)

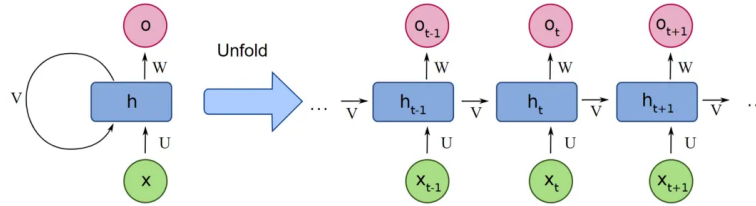


Figure 1.5: Recurrent Neural Networks

RNNs [Figure 1.5](#) model sequential dependencies by maintaining an internal hidden state $\mathbf{h}_t$ that summarizes information up to time t . The update rule typically is:

$$\mathbf{h}_t = f(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_x \mathbf{x}_t + \mathbf{b}) \quad (1.6)$$

where f is a nonlinear activation function (e.g., $\tanh$), $\mathbf{x}_t$ is the input at time t , and $\mathbf{W}_h, \mathbf{W}_x, \mathbf{b}$ are learnable parameters. This recurrent structure enables the network to capture temporal dependencies of arbitrary length.

In the case of RNNs, a different algorithm is applied to compute the gradients during optimization: Backpropagation Through Time (BPTT). It's the extension of the standard

backpropagation algorithm for training recurrent neural networks (RNNs). Unlike feed-forward neural networks, RNNs maintain a hidden state that evolves over time, making the gradient computation dependent on the temporal unfolding of the network.

To apply BPTT, the RNN is unrolled across a sequence of time steps, effectively transforming it into a deep feedforward network where each layer corresponds to the network at a specific time step. Given a sequence of inputs $(x_1, x_2, \dots, x_T)$ and the corresponding hidden states $(h_1, h_2, \dots, h_T)$, the loss is typically computed across all time steps as:

$$\mathcal{L} = \sum_{t=1}^T \ell(y_t, \hat{y}_t), \quad (1.7)$$

where $\hat{y}_t$ is the model output at time t , and ℓ is a suitable loss function.

Gradients are then backpropagated through the unrolled network, taking into account the recurrent connections between hidden states. Specifically, the gradient at time step t depends not only on the current error but also on gradients propagated from future time steps, due to the shared parameters across time. This accumulation is given by:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{t=1}^T \frac{\partial \mathcal{L}}{\partial h_t} \frac{\partial h_t}{\partial \theta}, \quad (1.8)$$

where θ represents the model parameters (e.g., weights and biases).

A key challenge in BPTT is the problem of vanishing and exploding gradients, especially when dealing with long sequences. This issue motivates the use of specialized architectures like LSTMs[8] and GRUs[9], which introduce gating mechanisms to better manage long-range dependencies.

In practice, a truncated version of BPTT is often used to limit computational cost, where the network is unrolled only for a fixed number of steps (e.g., k steps) instead of the full sequence length.

Attention Mechanisms

Attention [Figure 1.6](#) assigns dynamic weights to past inputs, allowing the model to focus on relevant parts of the input sequence when producing outputs. Given a query $\mathbf{q}$, keys

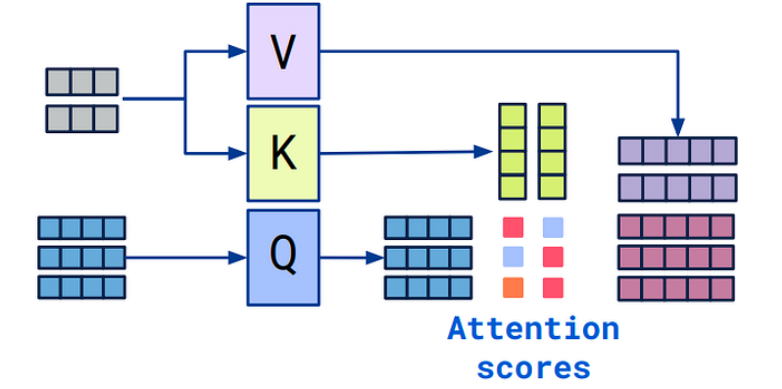


Figure 1.6: Attention mechanism

$\mathbf{k}_i$, and values $\mathbf{v}_i$, the attention output is computed as:

$$\text{Attention}(\mathbf{q}, \mathbf{K}, \mathbf{V}) = \sum_i \alpha_i \mathbf{v}_i, \quad \text{where} \quad \alpha_i = \frac{\exp(\mathbf{q}^\top \mathbf{k}_i)}{\sum_j \exp(\mathbf{q}^\top \mathbf{k}_j)} \quad (1.9)$$

A special case of this mechanism is **self-attention**, where the query, key, and value vectors are all derived from the same input sequence. This allows the model to relate each position in the sequence to all other positions, enabling a rich representation of contextual dependencies. Self-attention forms the core of the Transformer architecture and is defined as:

$$\text{SelfAttention}(\mathbf{X}) = \text{softmax} \left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}} \right) \mathbf{V} \quad (1.10)$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ are linear projections of the input $\mathbf{X}$, and d_k is the dimensionality of the keys. This weighted aggregation allows models such as Transformers to capture long-range dependencies efficiently.

While standard attention mechanisms scale quadratically with the sequence length due to the $\mathbf{Q}\mathbf{K}^\top$ computation, **linear attention** techniques have been proposed to address this limitation. These methods approximate or restructure the attention operation to achieve *linear complexity* in the input length. A typical formulation of linear attention rewrites the attention as:

$$\text{LinearAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \phi(\mathbf{Q}) \left(\phi(\mathbf{K})^\top \mathbf{V} \right) \quad (1.11)$$

where $\phi(\cdot)$ is a kernel feature map (e.g., ELU or ReLU-based) applied element-wise. This

factorization allows for more efficient computation, making linear attention attractive for long-sequence modeling tasks.

Attention mechanisms provide a flexible and powerful way to model dependencies, with self-attention enabling position-wise interactions and linear attention offering scalability to long sequences.

State Space Models (SSMs)

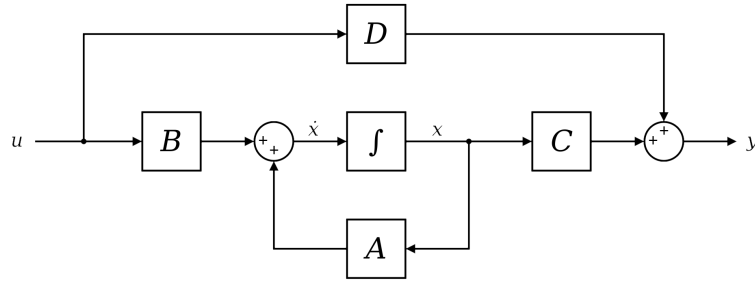


Figure 1.7: State Space Models

SSMs [Figure 1.7](#) describe the temporal evolution of a system through latent states $\mathbf{x}(t)$ driven by inputs $\mathbf{u}(t)$ [10]. The continuous-time linear SSM is defined by:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \quad \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \quad (1.12)$$

where:

- $\mathbf{x}(t) \in \mathbb{R}^n$ is the state vector representing the system's internal condition,
- $\mathbf{u}(t) \in \mathbb{R}^m$ is the input (control) vector,
- $\mathbf{y}(t) \in \mathbb{R}^p$ is the output vector,
- $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ are system matrices defining state transitions and output generation.

These models capture the dynamical behavior of the system and can be discretized for numerical implementation in time series forecasting tasks.

1.5 Time Series

Time series is a sequence of data points gathered at regular time intervals, representing a succession of observations for a phenomenon that changes over time[11]. From a

mathematical point of view, it is best described as data indexed or ordered by time.

Time series data tend to have a set of properties that differentiate it compared to regular tabular data:

- **Temporal order:** a lot of information is contained in the chronological order of the values, which makes it really important for the analysis.
- **Autocorrelation:** each value of the entry may be a correlation with past values, this indicates patterns over time.
- **Trend:** changes consistent over a long period of time.
- **Seasonality:** short-term periodic changes.
- **Outliers:** Abnormal data points that deviate from general patterns.
- **Irregularity:** Random fluctuations elements that are difficult to predict in time series data.
- **Cycles:** Longer period fluctuations that repeat over irregular time.
- **Non-Stationarity:** Statistical elements of time series data (such as mean and variance) may change over time, requiring specific transformations or modeling strategies.

1.6 Explainable AI (XAI)

In recent years, the rapid deployment of AI solutions in critical sectors has raised various concerns regarding its interpretability and transparency. Unlike traditional, easily interpretable models, modern architectures—especially those based on deep learning—are often perceived as black boxes. To bridge this interpretability gap, several explainability techniques have been developed[12]:

- **LIME (Local Interpretable Model-agnostic Explanations):**[13] It is a local surrogate method, meaning it approximates the model's prediction using weak learners in the neighborhood of the instance to explain the prediction, which enables it to work on all predictive models.
- **Shapley Values:**[14]At its core, it's a cooperative game theory method for assigning payouts to players depending on their contributions to the total payout. If

we apply it in our case, the players would be the features, the payouts would be the importance and the total payout would be the difference between the actual prediction and the average prediction. The Shapley value is the average marginal contribution of a feature value across all possible coalitions. In other terms, we average how much a given feature affects the prediction taking into consideration all possible combinations of sets of features excluding the feature in question.

Formally, the Shapley value for feature i is:

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} [f_{S \cup \{i\}}(x_{S \cup \{i\}}) - f_S(x_S)] \quad (1.13)$$

where N is the set of all features, S a subset excluding i , and f_S the model restricted to features in S .

This approach is robust and has been used in similar studies [15].

Conclusion

This chapter outlined the essential building blocks of Artificial Intelligence, tracing its evolution from rule-based systems to advanced machine learning and deep learning techniques. We highlighted the importance of understanding data characteristics such as time series properties and addressed the growing need for explainability in AI models to ensure transparency and trust. Equipped with this foundational knowledge, subsequent chapters will delve deeper into specific methods and applications relevant to this thesis.

Chapter 2

AI for HealthCare

AI is being increasingly used in healthcare to address challenges related to population health, patient experience, caregiver workload, and cost reduction. One of the earliest AI systems in medicine was CASNET (1978) [16], a rule-based diagnostic tool. Since then, advances in machine learning have allowed models to be trained on medical datasets, significantly expanding AI’s potential in the field.

2.1 Applications of AI in Healthcare

In recent years, AI has been extensively used in various aspects of the medical field[17]. In computer vision for example, many studies have achieved excellent results in diagnostic imaging with performance that’s comparable or even exceeds that of practitioners[18]. Other applications include AI-driven drug discovery[19], or in general, any combinatorial optimisation processes in healthcare, which enables researchers to better understand protein structures and develop more targeted therapeutics. We need also to mention the surge of AI Chatbot that have led to the deployment of many surprisingly reliable medical AI chatbots[20], or the newly adopted patient risk stratification [21] using machine learning.

The most relevant application to our study is vital sign monitoring[22], [23]. It uses algorithms to analyze subtle changes in biomarkers like sleep pattern, physical activity, heart rate or CGM signals to provide a more comprehensive view of the patient’s health. It enables early detection of health issues and improves predictive analytics for disease progression with limited or even non-existent downsides (unintended adverse events from

overtreatment and unnecessary diagnostic procedures).

2.2 Trustworthy AI in Healthcare

Regardless of its application, AI has proven its capacity to model complex medical interactions and successfully capture patterns needed for the prediction, but that alone is not enough to be applied in the medical field[24]. Given the unique nature of the healthcare sector, a specific set of regulations and guidelines ,in addition to high performance metrics ,has been established to ensure that AI models are safe, effective, and suitable for clinical use:

- **Explainability** : also called Interpretability or transparency, is to have access to the underlying logic behind the decisions taken by the model. This enables healthcare providers to judge the output of the model and decide whether to refer to it during their medical decisions.
- **Fairness** : Is to ensure that the output of the model is free from subjective or biased decisions, mainly talking about decisions based on protected and unrelated features or information (social background, ideological preferences, ...)
- **Privacy** : Medical AI models tend to have access to personal information about the patients, ensuring privacy and secrecy during the development and deployment of the model is crucial.
- **Robustness** : Making sure that the model’s decisions aren’t influenced or affected by unrelated third-parties.

These principles are essential to building AI tools that clinicians and patients can trust.

2.3 Related Work

Several studies have explored the prediction of glycemic events, especially hypoglycemia. While their goals are similar, methodologies differ significantly. Some rely on feature engineering, while others use time-series approaches that preserve temporal relationships.

In the feature engineering approach; authors of [25] included short, medium and long term features in addition to various demographic and snowball properties extracted from the vital signals. What was strange to us is the inclusion of contextual information such as hour and day in the input of the model. According to the feature importance of

the resulted Random Forest , the trained model put a lot of emphasis on the day and hour features (first and third most important feature respectively). Although this may improve the evaluation of the model, it will also limit it's ability to generalize to extraordinary situations where learned time patterns don't apply, an example for that would be a test population where meal time and lifestyle doesn't align with the training population customs. We also found that the authors didn't include metrics such as precision and F1-score, which (based on the shared confusion matrix) were of 49.7% and 65.5% respectively for 0-15 min predictions. Although we understand the reason behind putting more emphasis on the recall metric in the case of hypoglycemia predictions (hypoglycemia are far more dangerous to the patient life than hyperglycemia), we think that providing a predictive model that has more than 50% of false alerts will lead to the patient losing confidence in the solution. In the contextual variables, they used a linear function to calculate the insulin on board, which assumed insulin effects wear off in four hours. Carbohydrates were modeled as being absorbed at a rate of 0.5 g/min after an initial lag of 15 minutes.

For the time series approach; input data typically includes signals from CGM, heart rate, and other lifestyle factors. In our work, we focus on the univariate approach based solely on CGM signals and the multivariate approach that includes CGM data, meals, and bolus insulin. The minimal approach enhances practicality and improve the ease of use. Though some studies suggest additional features improve accuracy, we aim to evaluate whether simpler inputs can yield competitive performance.

In the paper [26], the authors opted for an event-based classification approach. They also proposed layered meta-learning approach, which uses a decision tree as a meta-learner to exploit the predictions from a multi-expert model based on either LSTM or CNN architectures. Each network is an expert in detecting one of the three investigated classes and pursues a univariate approach. They also implemented an on edge device based on their proposed pipeline. Although the study achieved remarkable results following their labeling policy (F1-score of 89.7% for hypoglycemia class), not only they didn't test how the resulted model performs on out of distribution settings, but the shared results can't be compared with the classical approaches of labeling. The authors didn't try to investigate the patterns learned by the deep learning models, no section for model explanation or interpretation was included.

Another study [27] targeted short term forecasting of glucose levels based on CGM signals only and using a deep learning architecture composed of Bi-LSTM and attention

layers. Unfortunately, the authors didn't precise the exact prediction horizon applied in the study, which makes comparison with other projects results impossible.

In the paper [28], authors made a head-to-head comparison between various linear and non-linear models. The results showed that the RMSE of the best linear model and the best non-linear model were similar in the regression task (22.15 and 21.52 mg/dL, respectively). Whereas in the classification task, the best linear algorithm (individualized ARIMA) provided precision = 64%, recall = 82%, which is better than the best nonlinear technique (population SVR): precision = 63%, recall = 69%. According to the authors, this has shown that non-linear algorithms do not provide any improvements compared to linear algorithms. We see that the results are only limited to the tested models in this specific experimental set-up, and doesn't generalize well to recent and up-to-date breakthroughs in deep learning or to real-world scenarios. We fully agree that more complex DL models aren't needed in every scenarios, and may seem to be overkill for some simple tasks. But considering the abundant amount of data available nowadays, modern deep learning models may be able to capture complex patterns that traditional linear models such as ARIMA can't detect. Another thing to point out, is that the tests didn't include any modern or recent non-linear models (Transformers, state space models, ...) which have proven their performance in sequence modeling. We also question the use of individualized CGM prediction models for real world scenarios. Although it may provide a boost in performance; due to the model being trained on personalized patterns; it rises some questions related to the ease of use, resiliency to lifestyle changes and the repeated training process. Not only will we need to collect CGM data for every new user, those training instances may become not representative due to a change in lifestyle or a change in medical situation. Training the model on a large population will make it more resilient to distribution shift [29].

In the article [30], authors investigated the importance of model explainability in the case of decision support systems (DSS) to forecast glucose levels and provide preventive therapeutic suggestions. They proposed using SHAP (SHapley Additive exPlanations) to explore the learned logic of black box models such as LSTM. The results have shown that a better performing model (RMSE: 21.43 mg/dl vs 21.67 mg/dl) doesn't imply a more physiologically sound model. By proposing corrective actions based on the models predictions, and evaluating it's correctness using a digital twin-based methodology[31], the authors showed that a physiologically sound model will lead to better correct suggestions even if it had a lower evaluation metric.

Based on the related articles viewed, we could distinguish two main predictive strategies: regression and classification.

Regression involves predicting upcoming CGM values using historical data. Approaches include:

- Autoregression: Predict one step at a time, feeding the prediction back into the model.
- Multi-step forecasting [32]: Predict multiple steps in one shot.
- Bayesian regression[33]: Estimate distributions instead of point values to capture uncertainty.

Autoregression is considered better in the case of a close prediction horizon (PH : interval to predict in the future), but its performance decreases significantly if the target variable is relatively far (needs a large number of steps to reach) due to errors stacking from previous predictions. Multi-step predictions tend to have better results for further targets, but suffer from high variance in the predictions. In addition to the previously described paradigms.

Classification to train a model to predict a label for a set of features. In this case, we will classify the sequence as positive if the upcoming measurements contain a value below the hypoglycemia threshold.

Unlike regression, there aren't various paradigms in this approach, the difference mainly lies in how labels are defined. In event-based predictions, we try to predict the event of hypoglycemia in the upcoming measures (interval of length PH), some consider it a positive class (Hypoglycemia) if the threshold is crossed during the interval, while others will label it positive if one or more value are under the hypoglycemia threshold (the main difference lies if crossing the threshold happened in the context introduced to the model or not). Another approach is to predict the probability of hypoglycemia for each upcoming time step, producing a sequence of risk values.

When it comes to models, various architectures were previously tested. The first papers focused on using Machine Learning algorithms solely, which didn't bring that good of a result. It was followed by the surge of memory based models (RNN and LSTM) and CNN models to extract temporal patterns in the signals. Some also updated their list of ML algorithms by applying modern one's like XGBoost. Finally, the most recent papers were more focused on using Attention based algorithms (Transformers) [33]–[35].

Another interesting aspect of hypoglycemia events prediction is highly imbalanced data, hypoglycemia events is a minority class and represents less than 5% of the dataset. Most of the published papers don't add any stage to their pipeline in order to bridge the gap between the occurrence of the different classes. Some made efforts to apply GANs (Generative Adversarial Networks) to CGM data for data augmentation purposes, while others limited themselves to using random noise in the training process to augment the input signal.

Table 2.1: Summary of Related Works on Glycemic Event Prediction

Ref	Approach	Model Type	Key Features	Limitations / Observations
[25]	Feature Engineering	Random Forest	Short, medium, long-term features; contextual (hour/-day), demographic	Overfit to time/context, poor generalization, no precision/F1 metrics, 49.7% precision, 65.5% F1; insulin/carbs modeled with simple rules
[26]	Event-based Classification	Meta-learning (LSTM, CNN, Decision Tree)	Expert models for each class, univariate CGM data, on-edge deployment	Good F1 (89.7%), but untested on out-of-distribution data; lacks model explainability; non-comparable labels
[27]	Time Series Forecasting	Bi-LSTM + Attention	CGM-only, DL architecture	No clear prediction horizon defined; limits reproducibility and comparison
[28]	Regression and Classification	ARIMA, SVR	Comparison of linear vs. non-linear models	Claims no benefit from non-linear models, but didn't test recent DL; lacks generalization to modern tasks
[30]	Explainability in Prediction	LSTM + SHAP	DSS context, digital twin validation	Better metrics don't imply better decision support; physiologically sound model more reliable

2.4 Problem Overview

Hypoglycemia, defined as a blood glucose level below 70 mg/dL, poses a significant risk to individuals with Type 1 Diabetes Mellitus (T1DM), potentially leading to severe complications such as seizures, loss of consciousness, or even death if not promptly addressed. Traditional self-monitoring methods, like finger-prick glucose meters, are invasive and offer only intermittent data, making them insufficient for timely detection and prevention of hypoglycemic events.

Continuous Glucose Monitoring (CGM) systems have emerged as a less invasive alternative, providing real-time glucose readings from interstitial fluid at five-minute intervals. However, these systems inherently lag behind actual blood glucose levels by approximately 5 to 20 min due to physiological delays in glucose diffusion. This latency can hinder timely interventions, as the CGM readings may not reflect rapid declines in blood

glucose levels promptly.

Our main goal is to enable preventive treatment of Hypoglycemia episodes through prediction of adverse glycemic events based on vital signs, mainly CGM signals. Taking into consideration that CGM signals tend to lag behind real blood glucose values for up to 15 min, we need a model that is able to predict hypoglycemia events at least 20 min before the event happens. We hypothesize that the time window of 5 min would be enough to take preventive measures, whether it is taking glucose or other aliment.

The critical challenge lies in developing a predictive model capable of forecasting hypoglycemic events at least 20 min in advance, thereby compensating for the CGM's inherent delay and allowing sufficient time for preventive measures, such as carbohydrate intake. This model must effectively analyze time-series data from CGM devices, accounting for the temporal dependencies and patterns indicative of impending hypoglycemia.

Moreover, the model must address the issue of data imbalance, as hypoglycemic events are relatively rare compared to readings in euglycemic range, which can bias predictive algorithms. Ensuring high precision and recall is paramount to minimize false alarms, which can lead to alarm fatigue and reduced trust in the monitoring system.

Therefore, the objective is to design a robust and reliable predictive model that utilizes CGM data to accurately forecast hypoglycemic events with a minimum lead time of 20 min. This model should enhance the safety and quality of life for individuals with T1DM by enabling timely interventions and reducing the incidence of severe hypoglycemic episodes.

Conclusion

AI has proven to be a transformative force in healthcare, particularly in chronic disease management. In this work, we explore its application in predicting hypoglycemic events for individuals with T1DM. By developing a trustworthy and efficient model based on minimal input data, we aim to enhance patient safety and enable early intervention. Our approach balances practicality with performance, contributing to more reliable AI tools in healthcare.

Chapter 3

Methodology

In this chapter, we will present different methods and techniques used in the proposed solution. We will start by introducing the general schema of the pipeline. Followed by an in-depth discussion of each stage : starting by preprocessing, data augmentation, modeling and various related technical decisions taken during the development.

3.1 Proposed pipeline

[Figure 3.1](#) contains the general schema for the proposed solution, starting by data extraction and parsing from datasets files, followed by preprocessing and data-augmentation, then model training and finally evaluation of the produced model. The evaluation was done with in-distribution settings, as well as out-of-distribution to ensure resilience and consistency. We tried to get an idea on how different approaches and techniques affect hypoglycemia prediction.

3.2 Preprocessing

Preprocessing refers to the transformations and operations applied to the data before it is used for analysis, modeling, or visualization. It aims to make the subsequent stages easier without affecting the data distribution, patterns, or contained information in a negative way. The different stages of the preprocessing pipeline can change drastically based on the type of data and the task to learn.

In time series analysis, the data consists of a collection of timed events [Figure 3.2](#).

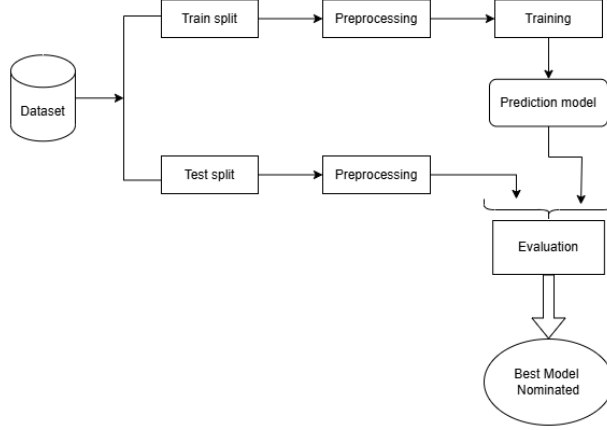


Figure 3.1: Proposed pipeline.

For our case, the timed events include the CGM value, the bolus dose, and meal kcal value. We organize the measurements into a set of sequences based on the CGM events, where each entry in the sequence is a CGM measurement accompanied by the bolus dose and the meal kcal value in the last 5 minutes (the timestep for CGM devices is 5 minutes). We aggregate (sum) each of the other events in the time interval while taking into consideration causality. If more than one consecutive null value is present in the segment, we divide the segment into two smaller sequences and apply the rest of the preprocessing stages independently.

To simulate insulin kinetics (as it does not take full effect instantly), we use the Insulin Action Profile (IAP) approximation. This method simulates the insulin activity over time following bolus injection. It typically involves a rapid onset, a peak effect, and a gradual decline. One common approach to approximate the insulin activity profile using mathematical functions is a piecewise linear function to simplify calculations (Onset Time: 0 min, Peak Time: 60 min, Duration: 240 min).

Another important design choice in time series problems is the look-back window (context) and the prediction horizon. This decision needs to achieve balance between improving user experience and maximizing performance. According to previous studies [36], the autocorrelation of consecutive CGM data points remains significant for approximately 1 hour, after which its significance drops below the level of 0.05. Based on that, we can say that CGM readings separated by more than 1 hour are linearly independent. In addition to that, the linear dependence of consecutive CGM readings disappears at

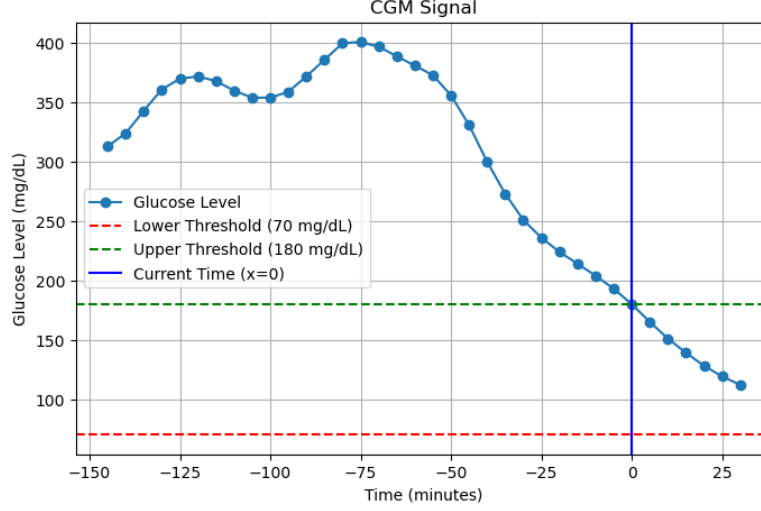


Figure 3.2: Example of a CGM signal.

a time lag around 30 min, therefore, a prediction of BG levels more than 30 min ahead would not be accurate using linear models. After the adequate values for the look-back window and prediction horizon are chosen, we use a sliding window to generate sequences of equal length, divide the resulting sequences into features and targets, and label the target sequences based on the measurements.

For the labeling policy we adopted an event-based approach, meaning that a glycemic adverse event would be detected if at least a single measurement is in the corresponding sequence. In the case of multiple adverse events coexisting with the same target sequence, we put higher priority for hypoglycemia as it is our main target, followed by hyperglycemia and finally euglycemia.

3.2.1 Data Augmentation

Adverse events in CGM signals are sparse compared to euglycemia. For hypoglycemia as an example, T1DM patients suffer on average two symptomatic episodes per week, if the episodes would last for 1 hour then less than 6% of the recorded CGM values are in hypoglycemia range. This leads to a severe imbalance in classes in the sequence classification task.

Various techniques may be employed to mitigate this inconvenience:

- **Cost Sensitive Learning:** A first simple solution would be to use cost sensitive

learning, in other terms, change the weights of the classes in the loss function. The lesser the occurrence the higher the class weight.

- **Addition of normal noise:** Another approach used by some previous studies [27] is adding normal noise $n \sim \mathcal{N}(0, \sigma^2)$ into the input sequence.
- **SMOTE** (Synthetic Minority Over-sampling Technique)[37]: the SMOTE algorithm helps generate synthetic instances in a less specific manner. The minority class is over-sampled by introducing synthetic examples along the line segments joining all or k of the minority class nearest neighbor.
- **Deep generative models:** Deep learning methods for synthetic data generation is achieved by three different architectures: Variational Auto-Encoders (VAE)[38], Generative Adversarial Networks (GAN) [39] and Denoising Diffusion Probabilistic Models (DDPM)[40]. In the context of CGM signals, to the best of our knowledge, only GANs were applied [41] even though Diffusion models have proven that they produce better quality and more varied results at the expense of performance [40]. As a consequence, we opted to apply Diffusion models on CGM signals for data augmentation.

3.3 Diffusion Models for data augmentation

Deep generative models for synthetic data generation can be achieved through various architectures, with Denoising Diffusion Probabilistic Models (DDPM) showing superior performance in generating high-quality synthetic data. In the context of CGM signals, we apply diffusion models for data augmentation to address class imbalance.

The principle behind diffusion models is to transform noise from a simple noise distribution to a data sample in the target distribution through a gradual denoising process. Mathematically, diffusion models are defined a forward process that gradually adds noise to data and a reverse process that learns to denoise.

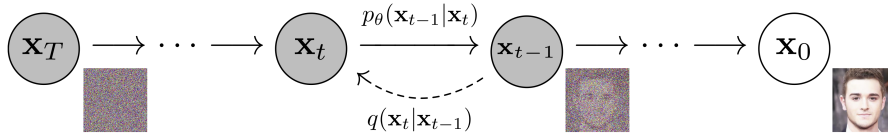


Figure 3.3: Forward and reverse process in DDPM

Forward Process: The forward process is a fixed Markov chain that gradually adds

Gaussian noise to the data according to a variance schedule $\beta_1, \beta_2, \dots, \beta_T$:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \quad (3.1)$$

Using the reparameterization trick, we can sample $\mathbf{x}_t$ at any timestep t directly:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I}) \quad (3.2)$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$.

Reverse Process: The reverse process is a learned Markov chain that starts from pure noise and gradually denoises to generate data:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)) \quad (3.3)$$

The training objective is to minimize the variational lower bound:

$$\mathcal{L} = \mathbb{E}_q \left[-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1) + \sum_{t=2}^T D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) \right] \quad (3.4)$$

In practice, this simplifies to a noise prediction objective:

$$\mathcal{L}_{simple} = \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}} \left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2 \right] \quad (3.5)$$

In the original work [42] (applied to image generation), the authors have chosen a U-net for the neural network that applies the reverse process, one of the reasons is to ensure that the output and input are of the same dimensionality. For signal generation, we only needed to change the convolutional layers from a 2d_Conv to 1d_Conv, and made some other adjustments to Attention layers to take into account the reduced dimensionality.

In the context of 1D diffusion models, the U-Net [42] architecture serves as the core denoising network that learns to reverse the forward diffusion process. Unlike traditional 2D image generation where U-Net operates on spatial dimensions, 1D diffusion models adapt the architecture to work with sequential data such as time series, audio signals, or any one-dimensional structured data.

3.3.1 Network Architecture

The U-Net for 1D diffusion [Figure 3.4](#) consists of an encoder-decoder architecture with skip connections, specifically designed to handle the denoising task at various noise levels determined by the diffusion timestep t .

Key Components

- **Encoder Path:** Progressively downsamples the input signal while increasing the feature dimension
- **Decoder Path:** Upsamples the encoded representation back to the original signal dimension
- **Skip Connections:** Preserve fine-grained information across resolution levels
- **Time Embedding:** Incorporates timestep information throughout the network
- **Attention Mechanisms:** Capture long-range dependencies in the signal

Input	Encoder	Middle	Decoder	Output
x_t ($C \times L$)	ResBlock + Attn ↓	ResBlock + Attn (bottleneck)	ResBlock + Attn ↑	$\hat{x}_0$ ($C \times L$)

Table 3.1: Components of U-net.

3.3.2 Mathematical Formulation

The U-Net learns to predict the noise ϵ added to the original signal $\mathbf{x}_0$ at timestep t :

$$\epsilon_{\theta}(\mathbf{x}_t, t) = \text{UNet}(\mathbf{x}_t, t) \quad (3.6)$$

where $\mathbf{x}_t$ is the noisy signal at timestep t , and θ represents the network parameters.

Noise Schedule

The forward diffusion process follows a linear beta schedule:

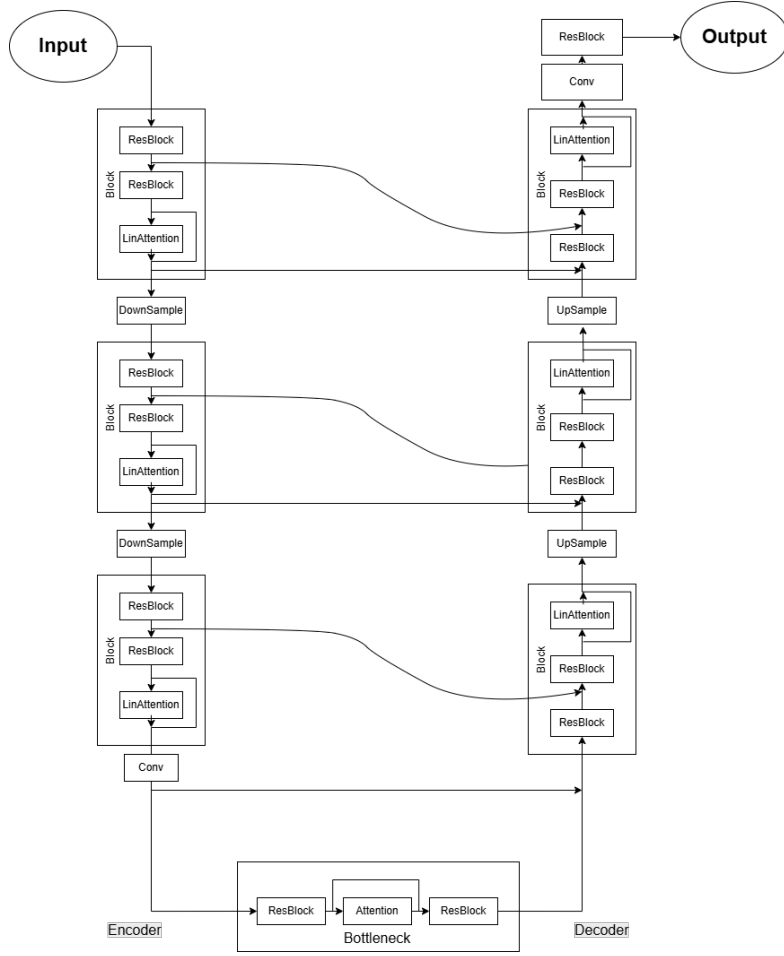


Figure 3.4: U-Net architecture for 1D diffusion models with encoder-decoder structure and skip connections.

$$\beta_t = \beta_{\text{start}} + \frac{t}{\text{timesteps}}(\beta_{\text{end}} - \beta_{\text{start}}) \quad (3.7)$$

$$\alpha_t = 1 - \beta_t \quad (3.8)$$

$$\bar{\alpha}_t = \prod_{s=1}^t \alpha_s \quad (3.9)$$

The noisy signal at timestep t is given by:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon} \quad (3.10)$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

3.3.3 Implementation Details

Network Configuration

For the 1D case, the U-Net is configured with:

- Input dimension equal to signal length L
- Channel dimension C (typically 1 for univariate signals)
- Dimension multipliers $(1, 2, 4)$ creating a 3-level hierarchy
- ResNet blocks with group normalization for stable training
- Linear attention mechanisms for computational efficiency

Time Embedding

The timestep t is embedded using sinusoidal position embeddings:

$$\text{PE}(t, 2i) = \sin\left(\frac{t}{10000^{2i/d}}\right), \quad \text{PE}(t, 2i + 1) = \cos\left(\frac{t}{10000^{2i/d}}\right) \quad (3.11)$$

This embedding is then processed through a multi-layer perceptron:

$$\mathbf{t}_{\text{emb}} = \text{MLP}(\text{PE}(t)) \in \mathbb{R}^{4d} \quad (3.12)$$

where d is the base dimension of the network.

3.3.4 Training Procedure

The model is trained using the simplified loss function:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}} \left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|^2 \right] \quad (3.13)$$

The training algorithm proceeds as follows:

Algorithm 1 Training U-Net for 1D Diffusion

```

1: Initialize U-Net parameters  $\theta$ 
2: Define noise schedule  $\{\beta_t\}_{t=1}^T$ 
3: while not converged do
4:   Sample batch  $\{\mathbf{x}_0^{(i)}\}_{i=1}^B$ 
5:   Sample timesteps  $t \sim \text{Uniform}(1, T)$ 
6:   Sample noise  $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
7:   Compute noisy signals  $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$ 
8:   Predict noise  $\hat{\epsilon} = \epsilon_\theta(\mathbf{x}_t, t)$ 
9:   Compute loss  $\mathcal{L} = \|\epsilon - \hat{\epsilon}\|^2$ 
10:  Update  $\theta$  using gradient descent
11: end while

```

3.3.5 Advantages for 1D Data

The U-Net architecture offers several advantages for 1D diffusion models:

- **Multi-scale Processing:** The hierarchical structure captures both local and global patterns in sequential data
- **Skip Connections:** Preserve fine-grained temporal information across different resolution levels
- **Conditional Generation:** Time embedding allows the network to adapt its behavior based on the noise level
- **Computational Efficiency:** Linear attention mechanisms reduce computational complexity for long sequences

This architecture has proven effective for various 1D generation tasks including time series synthesis, audio generation, and molecular sequence modeling.

3.4 Hyperparameter Tuning

Hyperparameter tuning is the process of searching for the optimal values of a machine learning model’s hyperparameters, which are the settings that define the model architecture and guide the learning process. This search process can lead to significant performance improvements, especially for complex models such as neural networks.

Mathematically, hyperparameter tuning involves finding a set of parameters Parm within a defined search space $\mathcal{H}$ (comprising all or a subset of possible hyperparameter combinations) that maximizes a performance criterion $f(\text{Parm})$:

$$\text{Parm}^* = \arg \max_{\text{Parm} \in \mathcal{H}} f(\text{Parm}) \quad (3.14)$$

A straightforward search strategy is **grid search**, where the model is trained using a predefined set of reasonable hyperparameter combinations, and the combination yielding the best performance is selected. While effective for simpler models, grid search becomes impractical for complex models due to the high computational cost associated with training each configuration.

An alternative is **random search**, where hyperparameter combinations are sampled randomly from the search space. Although it is more computationally efficient, random search does not focus the search around promising regions of the space, potentially missing optimal configurations.

In our proposed approach, we aim to balance search efficiency with execution time by employing a **genetic algorithm** to optimize model hyperparameters. Genetic algorithms are inspired by natural evolution and selection, and iteratively evolve a population of candidate solutions through biologically inspired operations.

3.4.1 Genetic Algorithm for Hyperparameter Optimization

The genetic algorithm proceeds through the following steps:

Algorithm 2 Genetic Algorithm for Hyperparameter Optimization

- 1: **Initialization:** initialize a population of individuals by randomly sampling from the hyperparameter search space.
 - 2: **while** termination condition not met **do**
 - 3: **Evaluation:** Evaluate the fitness of each individual in the current population.
 - 4: **Selection:** the best-performing individuals to act as parents.
 - 5: **Crossover:** Create offspring by combining selected parents based on their fitness.
 - 6: **Mutation:** Apply random mutations to offspring with a small probability to maintain diversity.
 - 7: Form the new population with the offspring and (optionally) elite individuals from the current generation.
 - 8: **end while**
 - 9: **Return** the best-performing individual found during the search.
-

The search process terminates when a maximum number of generations is reached or a convergence criterion is satisfied. This approach promotes exploration and exploitation of the search space, balancing performance with computational efficiency.

3.5 Models

We tested a wide range of models, from simple machine learning models to deep neural networks. Some of them include inductive bias (architectures designed to exploit sequential data), others act on different measures as different features without including the time ordering (mainly machine learning). Some would use positional embedding to add time information to the different measures embeddings, but we opted to limit the use of positional encoding to models that are inherently sequential. We need also to point out that all the mentioned models could be trained for classification or regression alike, especially when it comes to deep neural networks where the only difference could be limited to the change of the loss function and the final layer of each model.

3.5.1 Random Forest

Random forests [43] is a machine learning algorithm that uses an ensemble of decision trees trained on sub-samples of the dataset (bootstrap) and uses averaging to improve the predictive accuracy and control-overfitting. In the case of time series, random forests

treat data sequences as a single instance without taking into consideration the chronological information. Due to the fact that random forest only works on tabular data, we flattened the input signal when multiple channels are present (CGM data, approximation of carbohydrates intake, insulin dose). We mainly used random forests as a sanity check for more complex models.

3.5.2 XGBoost

XGBoost (eXtreme Gradient Boosting) [44] is a powerful and popular machine learning algorithm that uses gradient boosting on decision trees, making it well-suited for both regression and classification tasks. XGBoost uses a gradient boosting approach, which involves iteratively building a series of decision trees. Each tree is trained to correct the errors of previous trees, gradually increasing the model's performance. Like its machine learning model's counterpart, we flattened the input signal when multiple channels are present (CGM data, approximation of carbohydrates intake, insulin dose).

3.5.3 Long Short-Term Memory (LSTM)

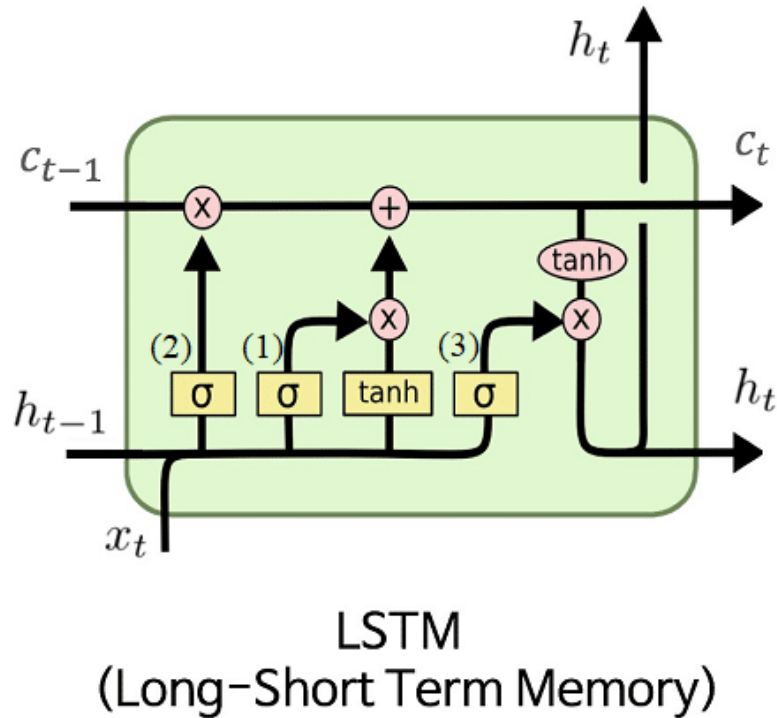


Figure 3.5: Long Short Term Memory cell.

LSTM [Figure 3.5](#) is a type of recurrent neural network (RNN) specifically designed to address the vanishing gradient problem in traditional RNNs, which enables it to learn long-term dependencies in sequential data. In our implementation, the LSTM model is composed of three main parts:

- **Embedding layer:** we use a linear layer to generate an embedding for each entry in our sequence, taking into consideration cases where the signal is composed of multiple channels.
- **LSTM cell:** the core of the model, it takes as an input the sequence of embeddings and iterates through it. The cell state is modified depending on the embeddings where it carries information across time steps. The output would be a latent vector that should contain the necessary information extracted from the sequence.
- **Decisional layer:** a linear layer to translate the latent vector from the hidden state dimension to the output space. The output space varies depending on the task, in the case of classification of three classes, the output will be a vector of three values.

Linear layers weights are initialized randomly by sampling from a normal distribution, whereas the biases are initialized to zero.

LSTM Cell

The Long Short-Term Memory (LSTM) cell addresses the vanishing gradient problem inherent in traditional recurrent neural networks by introducing a sophisticated gating mechanism that regulates information flow through time. The LSTM architecture, first proposed by Hochreiter and Schmidhuber (1997) [8], employs three primary gates—the forget gate, input gate, and output gate—alongside a cell state vector that serves as the network’s long-term memory component.

At time step t , the LSTM cell processes input $\mathbf{x}_t \in \mathbb{R}^d$ and the previous hidden state $\mathbf{h}_{t-1} \in \mathbb{R}^h$ to produce the current hidden state $\mathbf{h}_t$ and updated cell state $\mathbf{C}_t$. The computational sequence proceeds as follows:

The forget gate $\mathbf{f}_t$ determines which information from the previous cell state should be discarded:

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \quad (3.15)$$

The input gate $\mathbf{i}_t$ and candidate values $\tilde{\mathbf{C}}_t$ collectively regulate new information incorporation:

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \quad (3.16)$$

$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{W}_C \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_C) \quad (3.17)$$

The cell state update combines selective forgetting and information addition:

$$\mathbf{C}_t = \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{C}}_t \quad (3.18)$$

Finally, the output gate $\mathbf{o}_t$ controls the exposure of cell state information to the hidden state:

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \quad (3.19)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{C}_t) \quad (3.20)$$

where $\sigma(\cdot)$ denotes the sigmoid activation function, $\odot$ represents element-wise multiplication, $\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_C, \mathbf{W}_o$ are learnable weight matrices, and $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_C, \mathbf{b}_o$ are bias vectors. This architecture enables the LSTM to maintain gradient flow across extended temporal sequences while selectively preserving relevant information in its cell state memory.

Linear layers weights are initialized randomly by sampling from a normal distribution, whereas the biases are initialized to zero.

3.5.4 Convolutional Neural Network (CNN)

CNNs are a type of deep neural network that specializes in data with a spatial structure. Mainly used in Computer vision, it applies filters (also called kernels) on the input to extract spatial patterns. As 1-dimensional signals show patterns similarly to image data, CNNs could be applied to signals by only using 1-dimensional kernels. In our implementation, the CNN model is composed of following parts:

- **Convolutional block:** a single convolutional block is composed of three different layers. Firstly, the convolutional layer that extracts spatial features from the input by applying learned filters. We then apply an activation function (in our case Rectified Linear Unit “ReLU”) to add non-linearity. And finally we pass the output

through a Max Pool layer to reduce dimensionality. In our implementation, we opted for two of the described blocks with different numbers of input and output channels. At the end, we flatten the final result to make it ready for the MLP (reshape from batch size, channels, time-step $\rightarrow$ batch size, channels * time step).

- **Multi-Layer-Perceptron (MLP):** All the features extracted by the convolutional block will then be passed to the MLP in order to produce a decision in the target's dimension. We opted for an MLP with a single hidden layer and used the ReLU activation functions.

In the convolutional block, the chosen kernel was of size of 5, to enable the detection of larger patterns in the signal.

3.5.5 Inception-LSTM

It's Hybrid model that tries to take the best out of both worlds, using CNNs to extract spatial information and LSTM to extract temporal information. Compared to the generic LSTM model previously presented, we change the embedding section with a more complex convolutional layer. In our implementation, we decided to go for an Inception block as our spatial feature extractor:

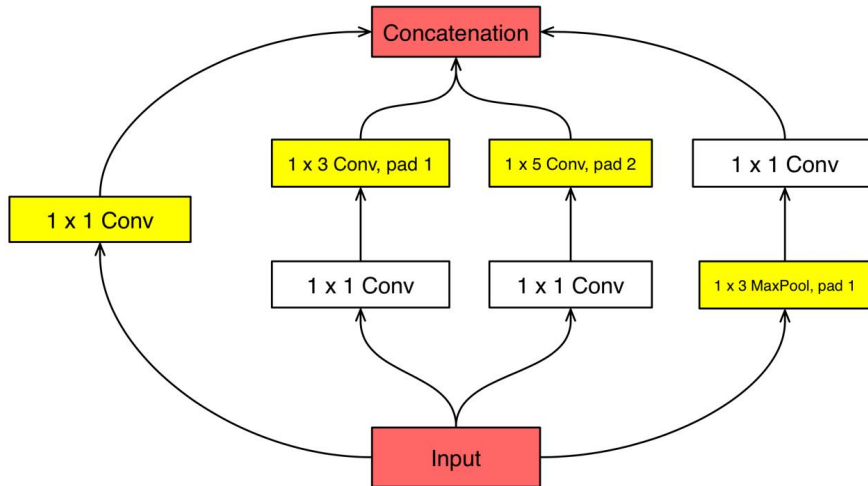


Figure 3.6: Inception Block.

- **Inception Block:** [45] the inception block Figure 3.6 consists of four parallel parallel branches with layers characterized by different kernel sizes (kernels of size 1,

3 and 5) to extract information from different spatial scales. Some branches add a kernel of size 1 to reduce the number of channels and decrease the complexity of the model. The last branch uses a size 3 max-pooling layer followed by a convolutional layer of size 1. All branches use appropriate padding to ensure the length of the sequence is consistent, and enable the concatenation of all feature maps. The hyperparameter for such a structure is the number of output channels per layer.

The initialization of the model’s weights and biases are handled based on the type of the layer:

- Linear Layers: weights are initialized through Xavier Uniform initialization, which is calculated based on the number of input and output features in order to improve the gradient flow during the forward pass and back-propagation. Biases are initialized to zero to provide a neutral starting point.
- 1D Convolutional Layers: weights are initialized using Kaiming Normal Initialization with the nonlinearity parameter set to “relu”. And biases are initialized to zero.

3.5.6 Transformer

Mainly developed for nlp tasks, Transformers are a type of neural network architecture for sequence learning. They use multiple layers of multi-head attention to capture dependencies of various ranges within a sequence. Unlike the original Transformer architecture proposed in the paper “Attention is all you need” [46] for machine translation task, our implementation leverges a decoder only structure with masked self-attention and processes input unidirectionally, similar to what’s found in GPT (Generative Pre-trained Transformer)[47] shown in Figure 3.7. Our implementation is built upon the following blocks:

- Embedding layer: we use a linear layer to generate embeddings for each entry in the sequence.
- Positional Embedding: we use positional embedding to add information related to the position of the measure in the sequence.
- Multi-Head Attention: this is the core component of the model. It takes as an input the sum of the embedding of each token and its positional encoding, and passes it through multiple layers of multi-head self attention to extract relationships between

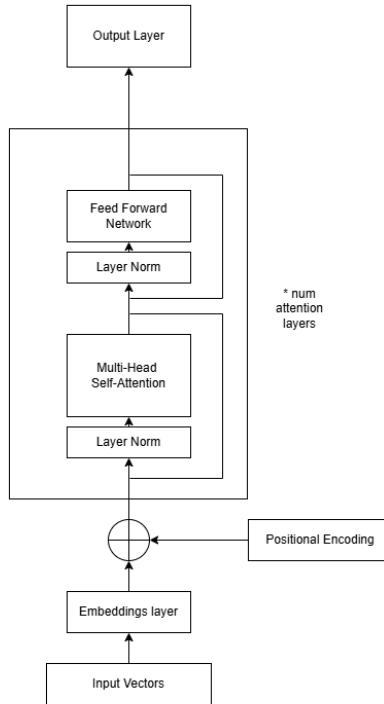


Figure 3.7: GPT-1 like model for time series.

different entries in the sequence.

- Output layer: we apply a layer normalisation to the output of the multi-head attention, followed by a linear layer to translate the latent vector to the output space. The prediction of the model would be in the last element according to the time-step axis, indicating that the model predicts the next token based on preceding tokens, a hallmark of autoregressive models such as GPT.

It's also important to describe the structure of the multi-head attention block:

- Each block is defined by: number of heads, embedding size, max context size.
- Each head takes as an input embeddings and output a latent vector of size equal to embedding size divided by number of heads, so that the concatenation of the outputs of all heads with result in a latent vector of the same shape as the input (embedding size).
- At the end of the layer, we apply a linear transformation accompanied with a

dropout layer to reduce overfitting.

We need to point out that we used two GPT variants, one uses linear layers to produce the Q , K , V matrices, while another one uses a convolutional layer. We referred to them as TRF and TRF-CNN respectively.

For the initialization, we used random values sampled from the normal distribution for the weights of linear layers and embeddings, and bias was initiated to zero.

3.5.7 S4 (Structured State Space Models)

According to the original authors [48], signal data such as voices, images and videos are discretization of continuous physical processes. And thus are ill suited for attention mechanisms, and need different architectures and approaches to better model it. The current self-attention used in transformer models could be described as a hard-attention. Means that we are trying to attend to a single token in the context to try and predict our next token, the only difference is that the token is an aggregation of the past sequence. This approach works well in natural language processing tasks where tokens are words with abstract meaning or concepts, but in the case of high dimensional signals such as images, it's not right to focus on a single pixel out of an image for the prediction. Another thing to note is that Attention-mechanism needs to store all the past context for the prediction, which is computationally inefficient. The best way to approach such high dimensional data is compression, where we build a latent representation that stores all the needed information of the context. For that purpose, the authors proposed S4, a highly efficient state space model based on the well-established theory of signal processing. Being built upon continuous-time state space models enable it to be particularly useful for handling continuous data:

- **Adaptability to Varying Sampling Rates:** The model can adapt to different sampling rates during inference by adjusting its internal step size.
- **Modeling Long-Range Dependencies:** S4 excels at capturing long-range dependencies in sequences, which is crucial for understanding patterns in continuous data over extended periods.
- **Efficient Computation:** Despite its capabilities, S4 maintains computational efficiency by leveraging multiple representations suitable for different situations, making it practical for real-world applications involving large-scale continuous data.

Continuous State Space Model: S4 [49] is based on the continuous-time state space representation:

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \quad (3.21)$$

$$y(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}u(t) \quad (3.22)$$

where $\mathbf{x}(t) \in \mathbb{R}^N$ is the state, $u(t) \in \mathbb{R}$ is the input, $y(t) \in \mathbb{R}$ is the output, and $\mathbf{A} \in \mathbb{R}^{N \times N}$, $\mathbf{B} \in \mathbb{R}^{N \times 1}$, $\mathbf{C} \in \mathbb{R}^{1 \times N}$, $\mathbf{D} \in \mathbb{R}$ are learned parameters.

In the case of deep learning SSMs, we could consider $\mathbf{D}$ a skip connection to lighten calculations. We will have :

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \quad (3.23)$$

$$y(t) = \mathbf{C}\mathbf{x}(t) \quad (3.24)$$

Discretization: To handle discrete sequences we use the trapezoid method, for a step size Δ :

$$\mathbf{x}_k = \overline{\mathbf{A}}\mathbf{x}_{k-1} + \overline{\mathbf{B}}u_k \quad (3.25)$$

$$y_k = \overline{\mathbf{C}}\mathbf{x}_k \quad (3.26)$$

where the discrete parameters are:

$$\begin{aligned} \overline{\mathbf{A}} &= \left(\mathbf{I} - \frac{\Delta}{2}\mathbf{A}\right)^{-1} \left(\mathbf{I} + \frac{\Delta}{2}\mathbf{A}\right) \\ \overline{\mathbf{B}} &= \left(\mathbf{I} - \frac{\Delta}{2}\mathbf{A}\right)^{-1} \Delta\mathbf{B} \\ \overline{\mathbf{C}} &= \mathbf{C} \end{aligned} \quad (3.27)$$

Different Computational Forms:

from the article [48], an SSM could have different views, each one of them has their advantages and disadvantages:

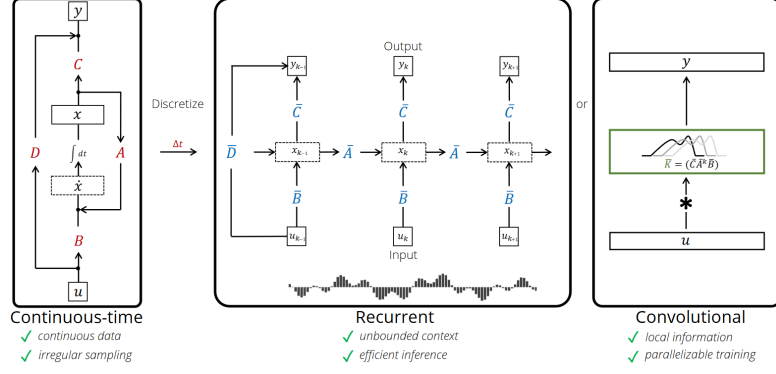


Figure 3.8: Different Computational Forms of S4.

1. *Recurrent Form:* The discrete state space model can be computed recurrently:

$$y_k = \bar{\mathbf{C}} \sum_{j=0}^k \bar{\mathbf{A}}^{k-j} \bar{\mathbf{B}} u_j \quad (3.28)$$

2. *Convolutional Form:* By defining the convolution kernel $\mathbf{K} \in \mathbb{R}^L$ where:

$$\mathbf{K}_j = \bar{\mathbf{C}} \bar{\mathbf{A}}^j \bar{\mathbf{B}}, \quad j = 0, 1, 2, \dots, L \quad (3.29)$$

The output becomes a convolution:

$$\mathbf{y} = \mathbf{K} * \mathbf{u} \quad (3.30)$$

This convolutional form enables efficient computation using FFT with $O(L \log L)$ complexity instead of $O(L^2)$ for the recurrent form.

HiPPO Initialization[50]: S4 uses the HiPPO (High-Order Polynomial Projection Operator) framework to initialize the matrix $\mathbf{A}$. HiPPO defines a principled way to compress continuous signals by projecting them onto orthogonal polynomial bases.

The HiPPO matrix for Legendre polynomials is:

$$\mathbf{A} = \begin{bmatrix} 1 & & & & & & & \\ -1 & 2 & & & & & & \\ 1 & -3 & 3 & & & & & \\ -1 & 3 & -5 & 4 & & & & \\ 1 & -3 & 5 & -7 & 5 & & & \\ -1 & 3 & -5 & 7 & -9 & 6 & & \\ 1 & -3 & 5 & -7 & 9 & -11 & 7 & \\ -1 & 3 & -5 & 7 & -9 & 11 & -13 & 8 \end{bmatrix} \Rightarrow \mathbf{A}_{nk} = \begin{cases} (-1)^{n-k}(2k+1), & \text{if } n > k \\ k+1, & \text{if } n = k \\ 0, & \text{if } n < k \end{cases} \quad (3.31)$$

Optimization Challenges and Solutions:

One of the main contributions presented in the S4 paper [49] is a stable method for speeding up the calculation of $\overline{\mathbf{A}}$ matrix power in the convolution filter Equation 3.29. To achieve that, authors focused on the special case of Diagonal Plus Low Rank (DLPR) SSM.

A DLPR SSM is $\mathbf{A} = \mathbf{P}\mathbf{Q}^*, \mathbf{B}, \mathbf{C}$, where $\mathbf{A}$ is diagonal and matrices $\mathbf{P}, \mathbf{Q}, \mathbf{B}, \mathbf{C} \in \mathbb{C}^{\times 1}$.

- **Transform to spectral domain:** Instead of computing the SSM convolution filter $\overline{\mathbf{K}}$ directly in time domain, we use the *z-transform* to introduce a generating function on its coefficients, after simplification, we can replace the sum of power operations with an inverse. We evaluate it at the roots of unity to get the filter's discrete Fourier and use the inverse Fourier transform to get back to the time domain.
- **Diagonal case:** to further speed-up the operation, we assume that matrix $\mathbf{A}$ is diagonal in order to compute the inverse faster than the naive inversion. Using Cauchy kernel (which is subject to many fast implementations), the authors replaced the inverse in the calculation of the *z-transform* by a weighted dot product.
- **Diagonal Plus Low Rank:** In order to relax the diagonal assumption, we assume that matrix $\mathbf{A}$ is of the form $\mathbf{A} = \mathbf{P}\mathbf{Q}^*, \mathbf{B}, \mathbf{C}$, where $\mathbf{A}$ is diagonal and matrices $\mathbf{P}, \mathbf{Q}, \mathbf{B}, \mathbf{C} \in \mathbb{C}^{\times 1}$.

$$(\mathbf{A} + \mathbf{P}\mathbf{Q}^*)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{P} \left(\mathbf{1} + \mathbf{Q}^*\mathbf{A}^{-1}\mathbf{P} \right)^{-1} \mathbf{Q}^*\mathbf{A}^{-1} \quad (3.32)$$

Using the Woodbury identity Equation 3.32 on the inverse operation, we are able to

write the z -transform of the filter as a sum of 4 weighted dot products:

$$\hat{K}_{DPLR}(z) = c(z) \left[k_{z,\Lambda}(\tilde{\mathbf{C}}, \mathbf{B}) - k_{z,\Lambda}(\tilde{\mathbf{C}}, \mathbf{P}) (1 + k_{z,\Lambda}(\mathbf{q}^*, \mathbf{P}))^{-1} k_{z,\Lambda}(\mathbf{q}^*, \mathbf{B}) \right] \quad (3.33)$$

We evaluate it at the roots of unity and apply the inverse Fourier transform to calculate the filter in time domain.

Advantages of S4:

- **Long-range dependencies:** Efficient modeling of very long sequences (up to 16K tokens)
- **Computational efficiency:** $O(L \log L)$ complexity through convolutional form
- **Adaptive sampling:** Can handle varying sampling rates during inference
- **Theoretical foundation:** Grounded in well-established signal processing theory
- **Memory efficiency:** Constant memory usage during inference unlike attention mechanisms

For our implementation, we replace the Multi-Head Attention in the Transformer architecture with S4 layers, maintaining a similar overall structure while leveraging the advantages of state space models for continuous signal processing.

3.5.8 Patching Technique

In all the previously presented deep learning models, we added a hyper parameter to control the patch size [Figure 3.9](#). In other words, we gave the possibility to introduce the signal of the measurements as a sequence of multi-channel measurements. The motive behind it is to include more semantic information in a single time-step, as a single measure lacks substantial information (they are often similar to their neighbouring values). So we opt to reshape the signal to a sequence of sequences, to enable the models to consider the broader context or patterns that may appear across multiple consecutive data points.

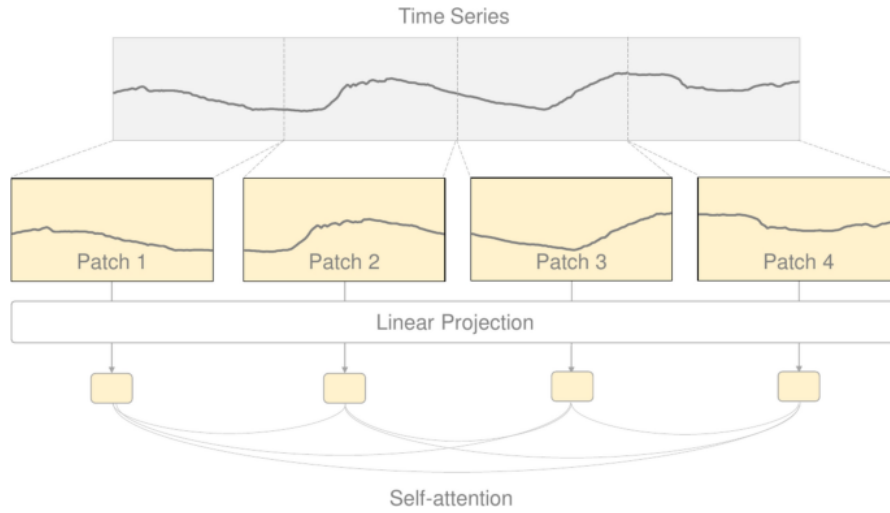


Figure 3.9: Patching technique in the case of Transformer

3.6 Design Choices

The task at hand could be solved using both a classification model or a regression model. From an optimization standpoint, using a sequence classification approach is more consistent, as we are more interested in the range of the value rather than the exact value of the upcoming measurement. But, from a practical point of view, and to better improve the patient's experience, providing a regression model may offer better insights on upcoming glycemic fluctuations and thus facilitate the treatment of the upcoming episodes. If classification models predict hypoglycemia, we won't have further information such as the estimated time or the degree of severity, which makes the treatment a bit hard.

As a result, we decided to train and evaluate on both approach as follows:

- Classification: we will follow an event based policy, where a sequence is labeled as hypoglycemia even if at least one of the upcoming values in the prediction horizon are under the threshold of 70 mg/dl. The look-back window is initially set to 100 mins whereas the prediction horizon is set to 10, 20 and 30 mins.
- Regression: we used multi-horizon prediction to predict all six measurements after each sequence. We also used autoregressive models when possible (the case of univariate time series)

Another important aspect is to compare the performance of univariate and multivariate time series:

- Multivariate: by providing additional information to the model in the form of insulin dosage and carbohydrate intake estimation, we could help it better understand the glucose levels dynamics. Especially when the provided factors are directly involved in glucose levels fluctuations.
- Univariate: even though limiting the input data might appear to be inefficient, once you take into consideration that patients will have to provide such information in real time to the predictor, you instantly foresee how this may affect the patients ease of use. It would become a side-hassle to introduce insulin dosage and an estimation of meal calories. And if wrong estimation is provided, the model results may become too far from real values. Thus it's better to limit the input data to CGM signals only.

We let the decisions on other technical parameters such as the inclusion of data augmentation, the loss function and the training hyper parameters based on performance metrics during the evaluation.

3.7 Training Set-Up

All models were built using default PyTorch modules, except for the S4 layer that needs custom layers. The training was performed on an Nvidia RTX 4060 GPU with 8GB of VRAM and 3072 cuda cores. We enable half-precision floating point (16-bit) training using PyTorch default function for autocasting, this decreased the memory and temporal complexity of operations by half. Maximum number of training epochs was set to 60, with an early stopping criteria set to 4 epochs. Train and test splits were done taking into consideration the chronological order and equal to 0.66 and 0.3 of the dataset, the rest was used for validation. if the train and test split was already defined by the dataset providers, we opted to use it. For all training processes, the optimizer used is Adam with learning rate that varies depending on the results of the hyper-parameter tuning.

Algorithm 3 Model Training with Optional Mixed Precision and Early Stopping

Require: Training data $(X_{\text{train}}, y_{\text{train}})$, Evaluation data $(X_{\text{eval}}, y_{\text{eval}})$, Number of epochs E , Early stopping patience p

- 1: Initialize model, optimizer, loss criterion
- 2: **if** preprocessor exists **then**
- 3: **if** preprocessor is not fit **then**
- 4: Fit preprocessor on X_{train}
- 5: **end if**
- 6: **end if**
- 7: Construct training and evaluation dataloaders
- 8: Initialize tracking variables: $L_{\text{train}}, L_{\text{eval}}, L_{\text{min}} \leftarrow \infty$
- 9: **for** $e = 1$ to E **do**
- 10: $L_{\text{epoch}} \leftarrow 0$
- 11: **for** each batch (X, y) in training data **do**
- 12: **if** preprocessor exists **then**
- 13: $X \leftarrow \text{preprocessor.transform}(X)$
- 14: **end if**
- 15: **if** fp16 is enabled **then**
- 16: Perform forward and backward pass using **mixed precision**
- 17: **else**
- 18: Perform standard forward and backward pass
- 19: **end if**
- 20: Accumulate training loss
- 21: **end for**
- 22: Append L_{epoch} to L_{train}
- 23: $L_{\text{eval_epoch}} \leftarrow 0$
- 24: **for** each batch (X, y) in evaluation data **do**
- 25: **if** preprocessor exists **then**
- 26: $X \leftarrow \text{preprocessor.transform}(X)$
- 27: **end if**
- 28: Perform forward pass only
- 29: Accumulate evaluation loss
- 30: **end for**
- 31: Append $L_{\text{eval_epoch}}$ to L_{eval}
- 32: **if** $L_{\text{eval_epoch}} < L_{\text{min}}$ **then**
- 33: $L_{\text{min}} \leftarrow L_{\text{eval_epoch}}$, reset patience counter
- 34: **else**
- 35: Decrement patience
- 36: **if** patience == 0 **then**
- 37: **return** $L_{\text{train}}, L_{\text{eval}}$
- 38: **end if**
- 39: **end if**
- 40: **end for**
- 41: **return** $L_{\text{train}}, L_{\text{eval}}$

3.7.1 Half-Precision Floating-Point Training in Deep Neural Networks

The adoption of 16-bit half-precision floating-point (FP16) arithmetic in deep neural network training has emerged as a critical optimization technique for modern large-scale models. FP16 representation, conforming to the IEEE 754-2008 standard, utilizes 1 bit for sign, 5 bits for exponent, and 10 bits for mantissa, yielding a significantly reduced memory footprint compared to standard 32-bit single-precision (FP32) arithmetic.

The primary advantages of FP16 training are multifaceted. Memory consumption reduction by approximately 50% enables training of substantially larger models within hardware constraints, as the memory requirement scales according to:

$$M_{\text{FP16}} = \frac{1}{2} \cdot M_{\text{FP32}} = \frac{1}{2} \sum_{l=1}^L (W_l + G_l + A_l) \quad (3.34)$$

where M denotes total memory usage, L represents the number of layers, and W_l , G_l , A_l correspond to weights, gradients, and activations for layer l , respectively. Additionally, modern GPU architectures, particularly those with Tensor Core units, demonstrate substantial computational throughput improvements, with theoretical speedups reaching $1.5\text{-}2\times$ for matrix operations.

However, FP16 training introduces significant numerical challenges that must be carefully addressed. The reduced precision range $[6.10 \times 10^{-5}, 65504]$ and limited mantissa precision can lead to gradient underflow, where small gradient values essential for convergence are truncated to zero. This phenomenon is particularly problematic during the initial training phases when gradients naturally exhibit small magnitudes.

The mathematical implications become apparent when considering the gradient update rule:

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_{\theta} \mathcal{L}(\theta_t) \quad (3.35)$$

where gradients $\nabla_{\theta} \mathcal{L}(\theta_t)$ may fall below the FP16 representable threshold $\epsilon_{\text{FP16}} = 2^{-24} \approx 5.96 \times 10^{-8}$.

To mitigate these limitations, mixed-precision training strategies have been developed. The most prevalent approach maintains FP32 precision for parameter updates while utilizing FP16 for forward and backward computations. Loss scaling techniques amplify

the loss function by a factor S before backpropagation:

$$\mathcal{L}_{\text{scaled}} = S \cdot \mathcal{L}(\theta) \quad (3.36)$$

$$\nabla_{\theta} \mathcal{L}_{\text{scaled}} = S \cdot \nabla_{\theta} \mathcal{L}(\theta) \quad (3.37)$$

subsequently rescaling gradients before parameter updates to prevent overflow while preserving small gradient information.

Dynamic loss scaling algorithms adaptively adjust S based on gradient statistics, typically increasing the scaling factor when no overflow occurs and reducing it upon detection of infinite or NaN values. This approach ensures numerical stability while maximizing the benefits of reduced precision arithmetic, enabling efficient training of contemporary transformer architectures and large language models.

3.8 Conclusion

This chapter presents a comprehensive methodology framework for CGM data analysis and glycemic event prediction. The combination of advanced preprocessing techniques, innovative data augmentation through diffusion models, and diverse model architectures provides a robust foundation for accurate prediction of glycemic events. The use of search algorithms for hyper parameter optimization will leverage the maximum potential of the proposed architectures. we also applied modern training techniques such as half-precision floating-point training to ensure computational efficiency.

Chapter 4

Evaluation, Results & Discussions

In this chapter, we conduct a thorough evaluation of the proposed solution. We begin by providing a detailed description of the datasets used in the study, including their structure, sources, and relevance to the problem at hand. This is followed by a definition of the performance metrics employed to assess the effectiveness of the models. We then present the experimental results obtained under different modeling approaches, highlighting key performance trends across various conditions. Finally, we discuss the main observations, interpret their implications in the context of the hypoglycemia prediction, and reflect on the strengths and limitations of the proposed approach.

4.1 Datasets

In medical settings, using different datasets for evaluation is critical to ensure the robustness, reliability and usefulness of predictive models:

- **Generalizability:** Medical data varies depending on region, equipment, patient demographics and patients. Testing on different datasets helps confirm that the model generalizes beyond the original distribution.
- **Bias Detection:** Medical datasets can carry hidden biases, such as overrepresentation of groups, ethnicities or disease severities. Using different datasets for evaluation may help detect if those biases were translated during the model's learning.
- **Robustness to Variability:** ensure that the predictive model performance is not

limited to the equipment used during data-collection. When it comes to CGM devices, many manufacturers provide different products with different specifications. We want to make sure that our predictive model will make reliable predictions regardless of the CGM used.

- External Validation: Clinical standards require out of distribution evaluation before regulatory approval or deployment.
- Overfitting Prevention: Make sure that the model is not learning random noise in the data but general principles.

For our study, we have chosen to use three different datasets:

The OhioT1DM Dataset[\[51\]](#)

The OhioT1DM Dataset comprises eight weeks of detailed physiological, behavioral, and glycemic data from 12 individuals with type 1 diabetes undergoing insulin pump therapy and continuous glucose monitoring (CGM). Participants used Medtronic 530G or 630G insulin pumps with Enlite CGM sensors and reported contextual life events via a smartphone app. Physiological data were collected using wearable devices: six participants wore the Basis Peak (2018 release), and six wore the Empatica Embrace (2020 release). The dataset includes CGM readings every five minutes, finger-stick glucose measurements, insulin dosages (basal and bolus), meal and carbohydrate intake, and self-reported events such as exercise, sleep, and illness. Sensor band data vary by device but include metrics such as heart rate, skin temperature, galvanic skin response (GSR), and physical activity. Each participant has two XML files—one for training/development and one for testing—totaling 24 files across the dataset. Tables in the original documentation provide participant demographics and data distribution.

The IOBP2 Dataset[\[52\]](#)

Entitled "The Insulin-Only Bionic Pancreas Pivotal Trial: Testing the iLet in Adults and Children with Type 1 Diabetes". This multi-center randomized controlled trial (RCT) evaluates the efficacy and safety of the insulin-only configuration of the iLet Bionic Pancreas (BP) System in managing type 1 diabetes (T1D) in both pediatric and adult populations. The study enrolls approximately 440 participants, including 165 children (aged between 6 and 18 years) and 275 adults (aged more than 18 years), across 16 clinical sites. Participants are randomized to BP or control groups for a 13-week intervention period, followed by a 2–4 day Transition Phase. The BP Group receives automated

insulin delivery via the iLet system, which integrates a touchscreen interface, infusion pump, and dosing algorithms informed by continuous glucose monitoring (CGM) from a Dexcom G6 sensor.

The primary objective is to assess superiority in HbA1c reduction at 13 weeks between BP and control groups. Key secondary endpoints include non-inferiority in time spent below 54 mg/dL and superiority across additional CGM metrics such as mean glucose, time in range (70–180 mg/dL), and glycemic variability measures. A hierarchical gate-keeping strategy is used to control type I error. Safety endpoints include the incidence of severe hypoglycemia, diabetic ketoacidosis, and other serious adverse events. Additional outcomes include quality of life and treatment satisfaction.

Randomization for pediatric participants follows a 2:1 ratio (BP vs. control), and for adults a 2:2:1 ratio (BP-lispro/aspart, BP-Fiasp, control). Following the RCT, participants in the BP arm undergo a transition to either their prior therapy or a therapy guided by the iLet system, randomized 1:1.

The DCLP3 Dataset[53]

The DCLP3 clinical protocol outlines a pivotal, randomized, controlled trial titled "The International Diabetes Closed Loop (iDCL) Trial: Pivotal Trial of t:slim X2 with Control-IQ Technology". The study evaluates the safety and efficacy of the t:slim X2 insulin pump integrated with Control-IQ Technology and Dexcom G6 continuous glucose monitoring (CGM) in individuals with type 1 diabetes. Conducted across seven U.S. clinical sites, the trial involves a 2:1 randomization of participants to either the closed-loop system (intervention) or a sensor-augmented pump (SAP) with no automated insulin delivery (control). The primary endpoint is the percentage of time spent in the target blood glucose range (70–180 mg/dL) over a 6-month period, assessed via CGM. Eligible participants are aged 14 and older, with type 1 diabetes for at least one year and using insulin therapy. Data collected include demographic and baseline characteristics, glycemic control metrics (HbA1c, time in range, mean glucose), insulin dosing data, device performance logs, safety outcomes (such as hypoglycemic and hyperglycemic events), and quality of life assessments through questionnaires. This study aims to generate evidence supporting the clinical benefits of automated insulin delivery systems for FDA premarket approval.

Aspect	OhioT1DM	IOBP2	DCLP3
Study Type	Observational	Multi-center RCT	Multi-center RCT
Participants	12 individuals with T1DM (7 M, 5 F)	440 (165 children, 275 adults)	168 individuals aged 14+ with T1DM
Device Used	Medtronic 530G/630G + Enlite CGM; Basis Peak or Empatica Embrace	iLet Bionic Pancreas + Dexcom G6 CGM	t:slim X2 + Control-IQ + Dexcom G6 CGM
Data Collected	CGM (5-min), insulin (basal/bolus), meals, exercise, sleep, illness, HR, GSR, temp	CGM, insulin dosing, clinical outcomes (HbA1c, time in range), adverse events	CGM, insulin dosing, time in range, HbA1c, adverse events, QoL
Duration	8 weeks per subject	13 weeks intervention + 2–4 day transition	6-month monitoring
Primary Endpoint	Blood Glucose Level Prediction	HbA1c reduction at 13 weeks	Time in range (70–180 mg/dL)
Extra Notes	Training/testing split per subject (2 XMLs each)	Pediatric and adult randomization, QoL, safety, glycemic metrics	FDA premarket evaluation; SAP used as control
Training instances (after applying preprocessing step)	98845	7663016	4775932
Proportion of hypoglycemia class (%) (after applying preprocessing step)	5.79	1.05	1.82

Table 4.1: Comparison of datasets used in the study.

4.2 Metrics of Evaluation

Even though our task could be achieved through different approaches, it is still mainly the prediction of Hypoglycemia events, and thus would be ultimately evaluated using classification metrics:

Confusion matrix: also known as an error matrix, is a table that visualizes the performance of a classification model by comparing its predictions to the actual results. It's a valuable tool for understanding where a model is making errors and for calculating performance metrics like accuracy, precision, and recall.

- **Accuracy:** it's the ratio of all classifications that were correct, whether positive or negative. Because it incorporates all outcomes of the confusion matrix, it could be used to describe the model's general performance in cases where the dataset is balanced. For heavily imbalanced datasets, if the positive class appears only 1% of the time, a model that always outputs negative class will get an accuracy of 99% despite being useless. That's why we rarely use accuracy as a performance metric in real-world cases.
- **Recall:** also called true positive rate (TPR), is the proportion of all actual positives that were classified correctly as positives. In imbalance datasets where the number of actual positives is very low, recall may be a better metric for evaluation than accuracy; it represents the ability of the model to detect positive instances.
- **Precision:** it's the proportion of all the model's positive classifications that are actually positive. It measures how much the model would output false positives. Precision and recall often show an inverse relationship, where improving one of them worsens the other.
- **F1-score:** is the harmonic mean of precision and recall. It gives a combined idea about precision and recall metrics. It is maximum when Precision is equal to recall, and thus is mostly used in cases where false positives and false negatives are equally costly.

In our use case, both false positives and false negatives are dangerous to the patients health. So we have chosen F1-score as a metric of evaluation when optimizing. But even though both situations are dangerous, from a physiological standpoint having a hyperglycemia event is less life-threatening than hypoglycemia events, we will then put more emphasis on recall of hypoglycemia while trying to keep the equilibrium between

the two rates.

For the evaluation of regression task, we used the following metrics:

- **Mean Absolute Error (MAE)**: it calculates the average of the absolute difference between the predicted and actual values, an intuitive measure for the average prediction measure.
- **Root Mean Squared Error (RMSE)**: its the square root of the average of the squared differences between predicted and actual values. Sensitive to outliers and penalizes larger errors, all while being intuitive by corresponding to the same units as the target variable.
- **R-squared** (Coefficient of determination): it measures the proportion of variance in the dependent variable that's explained by the model. Values range from 0 to 1; higher values indicate better fit.
- **Mean Absolute Percentage Error (MAPE)**: it expresses prediction errors as a percentage of actual values. Useful for understanding relative errors. But could become unstable for small target values, in the extreme case, not defined if $y = 0$.

In our use case, we want our regression model to be as consistent as possible, In other terms, have the least possible outliers in our predictions. Based on that, we argue that the most important metric would be the RMSE, as it is highly sensitive to big differences due to the squaring.

4.3 Evaluation

In the evaluation section of this work, we will present the evaluation of the previously described models. All model hyper parameters were tuned and trained on the Ohio T1DM dataset. We will start with the multivariate time series:

4.3.1 Multivariate

We start our test with the multivariate approach, the model will take as an input the CGM measurements, meal info and insulin dosage.

Classification

The following evaluation is for classification models for binary classification and multi-class classification:

The [Table 4.2](#) contains results of evaluation of previously presented models. The models were trained on classification task based on a multivariate input signal that contains CGM measures, the patient’s carbohydrate estimate of taken meal and insulin dosages. At first glance, we observe that the overall performance of all models decrease with the increase of the prediction horizon, we explain this by the introduction of various uncertainties the further we try to predict. Those uncertainties aren’t part of the context presented to the model, which makes it impossible to predict. Another observation is related to the models overall performance for each prediction range, the accuracy is consistent across all trained models. When it comes to our target class (Hypoglycemia), the most if not all recorded F1-scores tend to be close in one prediction horizon range (PH 10 min : 0.87, PH 20 min : 0.82, PH 30 min : 0.77). What really fluctuates is the precision-recall trade-off. For PH = 10, 20 min, the highest precision for the target class was achieved by CNN accompanied with a remarkably lower recall, For PH = 30 min the S4 model had a precision score of 0.899 with a recall as low as 0.674. When it comes to recall, TRF-CNN had one of the highest recalls at all ranges (at order: 0.928, 0.819, 0.797) accompanied with a lower than average precision (at order: 0.829, 0.83, 0.762). The best F1-score at PH 10 min is the RF (0.891), for PH = 20, 30 min it was the TRF(0.826) and LSTM (0.793) respectively. The models that offered the most balanced and consistent performance were RF and LSTM, although RF struggled at longer ranges (considerably lower recall of 0.724), this could be related to it not being able to learn complex patterns needed at longer ranges. Another interesting observations, is that RF algorithm performed slightly better than XGB in all scenarios, based on that we could assume that XGB is suffering from overfitting, as it’s has more modeling capability’s compared to RF, but also lack the inductive bias included in other sequential models.

Table 4.2: Binary Classification evaluation for Different Prediction Horizons: Multivariate approach.

PH (min)	Model	ACC	Hypo			Non-Hypo		
			Prec	Rec	F1	Prec	Rec	F1
10	RF	0.993	0.911	0.872	0.891	0.995	0.997	0.996
	XGB	0.992	0.890	0.875	0.882	0.996	0.996	0.996
	LSTM	0.992	0.851	0.911	0.880	0.997	0.994	0.996
	CNN	0.991	0.935	0.786	0.854	0.993	0.998	0.995
	HYB	0.991	0.847	0.874	0.860	0.996	0.995	0.995
	TRF	0.991	0.829	0.925	0.875	0.997	0.993	0.995
	TRF-CNN	0.991	0.829	0.928	0.876	0.997	0.993	0.995
	S4	0.991	0.829	0.909	0.867	0.997	0.994	0.995
20	RF	0.987	0.868	0.780	0.822	0.991	0.995	0.993
	XGB	0.987	0.847	0.792	0.819	0.992	0.994	0.993
	LSTM	0.988	0.836	0.828	0.832	0.993	0.993	0.993
	CNN	0.981	0.975	0.502	0.663	0.981	0.999	0.990
	HYB	0.927	0.899	0.749	0.817	0.990	0.996	0.993
	TRF	0.988	0.886	0.774	0.826	0.991	0.996	0.993
	TRF-CNN	0.987	0.830	0.819	0.824	0.993	0.993	0.993
	S4	0.986	0.897	0.708	0.792	0.989	0.996	0.993
30	RF	0.982	0.829	0.724	0.773	0.988	0.993	0.990
	XGB	0.981	0.800	0.721	0.759	0.988	0.992	0.990
	LSTM	0.984	0.842	0.750	0.793	0.989	0.993	0.991
	CNN	0.980	0.755	0.779	0.767	0.990	0.989	0.989
	HYB	0.981	0.766	0.790	0.778	0.991	0.989	0.990
	TRF	0.982	0.788	0.781	0.784	0.990	0.991	0.990
	TRF-CNN	0.981	0.762	0.797	0.779	0.991	0.989	0.990
	S4	0.983	0.899	0.674	0.771	0.986	0.996	0.991

Table 4.3: Three-Class Classification evaluation for Different Prediction Horizons: Multivariate approach.

PH (min)	Model	ACC	Hypo			Eu			Hype		
			Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
10	RF	0.967	0.903	0.860	0.881	0.973	0.974	0.974	0.962	0.964	0.963
	XGB	0.969	0.896	0.875	0.885	0.976	0.974	0.975	0.964	0.968	0.966
	LSTM	0.970	0.875	0.914	0.894	0.978	0.975	0.976	0.967	0.968	0.967
	CNN	0.934	0.828	0.913	0.868	0.913	0.987	0.949	0.994	0.839	0.910
	HYB	0.957	0.820	0.915	0.865	0.953	0.979	0.966	0.980	0.922	0.950
	TRF	0.969	0.875	0.909	0.892	0.978	0.972	0.975	0.962	0.970	0.966
	TRF-CNN	0.966	0.837	0.927	0.880	0.985	0.959	0.972	0.945	0.981	0.963
	S4	0.968	0.937	0.794	0.860	0.968	0.981	0.974	0.970	0.960	0.965
20	RF	0.948	0.860	0.778	0.817	0.961	0.957	0.959	0.931	0.950	0.940
	XGB	0.952	0.866	0.796	0.829	0.965	0.959	0.962	0.935	0.955	0.944
	LSTM	0.952	0.820	0.823	0.821	0.972	0.953	0.962	0.930	0.966	0.948
	CNN	0.945	0.932	0.667	0.778	0.969	0.944	0.956	0.903	0.977	0.939
	HYB	0.942	0.911	0.689	0.784	0.971	0.937	0.954	0.895	0.981	0.936
	TRF	0.951	0.860	0.793	0.825	0.968	0.955	0.961	0.929	0.961	0.945
	TRF-CNN	0.948	0.797	0.842	0.819	0.979	0.939	0.958	0.911	0.978	0.943
	S4	0.948	0.886	0.743	0.808	0.951	0.968	0.959	0.947	0.931	0.939
30	RF	0.928	0.835	0.713	0.769	0.946	0.943	0.944	0.902	0.924	0.913
	XGB	0.935	0.828	0.722	0.772	0.954	0.945	0.949	0.907	0.941	0.924
	LSTM	0.934	0.780	0.796	0.788	0.951	0.947	0.949	0.919	0.924	0.921
	CNN	0.928	0.914	0.588	0.716	0.954	0.934	0.944	0.878	0.960	0.917
	HYB	0.924	0.709	0.826	0.763	0.948	0.935	0.941	0.908	0.914	0.911
	TRF	0.931	0.822	0.760	0.790	0.941	0.953	0.947	0.922	0.905	0.914
	TRF-CNN	0.913	0.680	0.855	0.758	0.918	0.951	0.934	0.944	0.841	0.889
	S4	0.930	0.895	0.668	0.765	0.939	0.955	0.947	0.915	0.912	0.914

The [Table 4.3](#) contains the results of evaluation of multi-classes. Unlike the case of binary classification, the accuracy scores show a wider spread in accuracy values between models, highlighting the greater complexity of the task. This inconsistency may be a direct result of introducing a new class to targets.

To our surprise, there wasn't a notable change in the overall performance of most models compared to Binary classification. In some situations, we observe that the precision-recall tradeoff was redistributed for some models (CNN scored one of the highest recalls 0.913 at PH : 10 min). The TRF-CNN model still offers the highest recall across all ranges (at order:0.928 0.842, 0.855). And TRF and XGB had the most consistent performance across all ranges and a well balanced precision-recall trade-off. In this evaluation

XGB surpassed RF in all evaluations. For hypoglycemia predictions, we favor the LSTM model as it got consistent performance and well balanced precision-recall trade-off in addition to providing above average recall scores.

Regression

The following evaluation is for regression models:

Model	MAE	RMSE	MAPE	R^2	F1@10	F1@20	F1@30
RF	8.36	13.28	5.67	0.951	0.890	0.828	0.778
XGB	8.48	13.45	5.73	0.950	0.887	0.820	0.762
CNN	9.65	14.32	6.24	0.943	0.858	0.824	0.788
HYB	9.43	14.96	6.22	0.938	0.865	0.800	0.747
TRF	8.93	14.03	6.01	0.945	0.853	0.785	0.730
TRF-CNN	10.33	15.08	7.08	0.937	0.855	0.787	0.730
S4	8.27	13.14	5.60	0.952	0.890	0.825	0.775

Table 4.4: Regression performance metrics and hypoglycemia F1-scores across prediction horizons (10, 20, 30 mins): Multivariate approach.

Table 4.4 shows regression results of the proposed models. The S4 model achieved best scores in all regression metrics (more importantly RMSE: 13.14 mg/dl), the second best performer was RF (RMSE: 13.28 mg/dl). In the classification of the predicted measures, The RF model surpassed the S4 as the best performer at PH = 10, 20 min. Unlike regression, the best F1@30 (0.788) was achieved by the CNN model. This shows that regression performance doesn't translate directly to the regression task, especially at longer ranges. All models, except CNN, had notably higher precision and lower recalls. The precision-recall trade-off for CNN model was well balanced (PRC: 0.796, REC: 0.780).

4.3.2 Univariate

The following results are the evaluation of models trained on CGM signals only:

Classification

The following evaluation is for classification models for binary classification and multi-class classification:

Table 4.5: Binary Classification evaluation (Hypo, Eu, Hype) for Different Prediction Horizons: Univariate approach.

PH (min)	Model	ACC	Hypo			Non-Hypo		
			Prec	Rec	F1	Prec	Rec	F1
10	RF	0.993	0.910	0.889	0.899	0.996	0.997	0.996
	XGB	0.992	0.889	0.882	0.885	0.996	0.996	0.996
	LSTM	0.992	0.836	0.922	0.877	0.997	0.994	0.995
	CNN	0.986	0.717	0.907	0.801	0.997	0.988	0.992
	HYB	0.992	0.942	0.816	0.875	0.994	0.998	0.996
	TRF	0.993	0.899	0.882	0.890	0.996	0.996	0.996
	TRF-CNN	0.992	0.864	0.906	0.885	0.997	0.995	0.996
	S4	0.991	0.821	0.916	0.866	0.997	0.993	0.995
20	RF	0.988	0.863	0.803	0.832	0.992	0.995	0.993
	XGB	0.987	0.850	0.799	0.823	0.992	0.994	0.993
	LSTM	0.988	0.913	0.743	0.819	0.990	0.997	0.993
	CNN	0.980	0.885	0.538	0.669	0.983	0.997	0.990
	HYB	0.987	0.821	0.816	0.819	0.993	0.993	0.993
	TRF	0.988	0.899	0.755	0.821	0.990	0.996	0.993
	TRF-CNN	0.987	0.855	0.799	0.826	0.992	0.994	0.993
	S4	0.987	0.835	0.821	0.828	0.993	0.993	0.993
30	RF	0.982	0.828	0.736	0.780	0.988	0.993	0.991
	XGB	0.982	0.819	0.737	0.776	0.988	0.993	0.990
	LSTM	0.983	0.877	0.683	0.768	0.986	0.995	0.991
	CNN	0.973	0.927	0.387	0.546	0.974	0.998	0.986
	HYB	0.980	0.850	0.633	0.726	0.938	0.935	0.937
	TRF	0.982	0.804	0.767	0.785	0.990	0.992	0.991
	TRF-CNN	0.983	0.858	0.711	0.778	0.987	0.995	0.943
	S4	0.982	0.839	0.718	0.774	0.988	0.994	0.991

The [Table 4.5](#) contains evaluation of proposed models in the binary classification task based on CGM signal only. Like previous entries, the overall performance decreases with the increase of prediction range, and the spread of global metrics such as accuracy is small.

At PH:10 min, the HYB model achieved best precision of 0.942 at the expense of a low recall of 0.816. Highest recall of 0.922 was achieved by LSTM model,, while the RF had the best F1-score of 0.899.

At PH:20 min, LSTM model changed behavior and achieved highest precision score of 0.913, while S4 had the highest recall 0.821. Like the previous prediction horizon, RF had the best F1-score of 0.832.

At PH:30 min, minor changes appeared in the general patterns previously observed. Highest precision was achieved by CNN, highest recall by TRF and the best F1-score by RF.

Unlike the multivariate case, we noticed that models had considerably lower recall in the univariate approach, although no major difference was detected on the F1-scores. we could assume that the information provided by the insulin dosage contribute to the prediction of hypoglycemia. Another thing to note out is the decrease of the CNN performance overall.

Table 4.6: Three-Class Classification Metrics (Hypo, Eu, Hype) for Different Prediction Horizons: Univariate approach.

PH (min)	Model	ACC	Hypo			Eu			Hype		
			Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
10	RF	0.969	0.905	0.885	0.895	0.975	0.975	0.975	0.964	0.966	0.965
	XGB	0.969	0.898	0.875	0.887	0.976	0.974	0.975	0.963	0.968	0.966
	LSTM	0.968	0.911	0.888	0.899	0.966	0.983	0.974	0.977	0.948	0.962
	CNN	0.954	0.877	0.791	0.832	0.966	0.959	0.962	0.937	0.959	0.948
	HYB	0.966	0.885	0.900	0.892	0.965	0.981	0.973	0.976	0.946	0.961
	TRF	0.968	0.882	0.895	0.888	0.977	0.971	0.974	0.960	0.969	0.964
	TRF-CNN	0.959	0.795	0.931	0.858	0.957	0.978	0.967	0.981	0.927	0.954
	S4	0.958	0.959	0.746	0.839	0.949	0.987	0.967	0.978	0.927	0.952
20	RF	0.951	0.868	0.797	0.831	0.964	0.959	0.961	0.934	0.953	0.944
	XGB	0.952	0.862	0.792	0.825	0.966	0.958	0.962	0.935	0.957	0.946
	LSTM	0.954	0.863	0.805	0.833	0.972	0.955	0.963	0.930	0.967	0.948
	CNN	0.913	0.840	0.632	0.722	0.966	0.896	0.929	0.837	0.979	0.902
	HYB	0.949	0.969	0.607	0.746	0.958	0.963	0.960	0.932	0.961	0.946
	TRF	0.945	0.815	0.843	0.829	0.948	0.967	0.957	0.954	0.913	0.933
	TRF-CNN	0.949	0.829	0.817	0.823	0.975	0.943	0.959	0.913	0.974	0.943
	S4	0.948	0.838	0.807	0.822	0.963	0.955	0.959	0.932	0.950	0.941
30	RF	0.932	0.833	0.742	0.785	0.951	0.944	0.948	0.905	0.932	0.918
	XGB	0.934	0.822	0.729	0.773	0.955	0.943	0.949	0.905	0.941	0.923
	LSTM	0.936	0.824	0.758	0.790	0.967	0.933	0.950	0.890	0.966	0.927
	CNN	0.920	0.750	0.709	0.729	0.947	0.929	0.938	0.887	0.928	0.907
	HYB	0.926	0.843	0.723	0.778	0.930	0.959	0.944	0.928	0.885	0.906
	TRF	0.923	0.755	0.801	0.777	0.929	0.954	0.942	0.934	0.873	0.902
	TRF-CNN	0.930	0.779	0.7849	0.782	0.953	0.939	0.946	0.904	0.931	0.917
	S4	0.924	0.812	0.765	0.788	0.925	0.961	0.943	0.936	0.868	0.920

The [Table 4.6](#) contains results of the evaluation of the proposed models in classification task based on CGM signal. The major dynamics still hold in this approach, where models performance varies depending on the prediction range. For PH of 10 min, the best performer is LSTM model (PRC: 0.911, REC: 0.888, F1: 0.899), if we put more emphases on recall, we would nominate the HYB model (PRC: 0.885 , REC: 0.9, F1: 0.892) which was the third overall best performer based on F1-score.

At longer ranges, all trained models had precision scores better than recall scores. Except for the TRF, TRF-CNN and S4 models that had not only good overall performance; but also higher recalls accompanied with good precision scores.

Regression

The following evaluation is for autoregression and multi-horizon models :

Table 4.7: Regression Performance for Glucose Prediction (Autoregressive and Multi-Horizon Settings).

Setting	Model	MAE	RMSE	MAPE	R^2	Hypo F1@10	Hypo F1@20	Hypo F1@30
Autoregressive	RF	8.62	13.77	5.82	0.947	0.897	0.840	0.788
	XGB	8.92	14.17	5.85	0.944	0.880	0.823	0.776
	CNN	17.67	23.77	10.79	0.845	0.783	0.729	0.651
	HYB	14.11	19.50	11.20	0.895	0.806	0.735	0.676
	TRF	8.83	13.91	5.96	0.946	0.894	0.822	0.767
	TRF-CNN	9.23	14.78	6.07	0.940	0.888	0.817	0.762
	S4	23.48	31.92	14.24	0.720	0.884	0.815	0.759
Multi-Horizon	RF	8.49	13.44	5.78	0.950	0.885	0.825	0.776
	XGB	8.592	13.64	5.82	0.948	0.882	0.824	0.767
	CNN	10.17	15.11	6.97	0.937	0.835	0.775	0.724
	HYB	9.86	14.69	6.32	0.940	0.867	0.822	0.776
	TRF	8.428	13.42	5.62	0.950	0.879	0.813	0.754
	TRF-CNN	8.55	14.01	5.60	0.946	0.898	0.832	0.780
	S4	8.63	13.57	5.67	0.949	0.874	0.831	0.786

When it comes to the univariate regression task [Table 4.7](#), we could discern two approaches: auto-regression and multi-horizon regression.

At first glance, we clearly observe that all models performed better in the multi-horizon approach. The best resulted regressor was the RF with a RMSE of 13.44 mg/dl. In the other hand, the best F1-score@30 (0.788) was achieved by the RF using the autoregression approach.

Similar to the multivariate case, the univariate classification based on time series forecasting suffered from heavily unbalanced precision-recall trade-off (low recalls and very high precision).

Observations

To our surprise, there wasn't a major difference between different approaches for hypoglycemia events predictions. The global metrics tend to be consistent across the board, while class wise metrics change tendencies between different approaches. As a result of recall/precision trade-off, some algorithms showed a tendency to score better results according to one metric at the expense of the other (Hybrid models tend to have higher precision).

When it comes to overall consistency of performance, ML algorithms (XGBoost and Random Forest) were clearly superior. While DL models were less consistent, it could be a result of not tuning the hyperparameters for each task and range. In the classification task, the best overall performer of the DL algorithms was LSTM, followed by Transformer-CNN

Despite the ML algorithms showing more consistent results across different horizon-prediction, DL models tend to have the edge and show better scores in both multivariate and univariate classification.

Adding channels with information about meals and insulin dosage didn't bring any major improvement in classification other than improving precision-recall trade-off in the case of binary classification.

The binary classification didn't provide any improvement compared to the multi-class classification. In both cases (uni/multi-variate), the multi-class approach held better results than the binary one.

Overall, hypoglycemia class F1-score hovers around 0.87 for prediction horizon set to 10

min, 0.82 for prediction horizon set to 20 min and 0.77 for prediction horizon set to 30 min. Best recall was recorded at 30 min prediction horizon is 0.855 by TRF-CNN model at the expense of a precision 0.680, meaning that more than 85% of the hypoglycemia events were detected.

In the regression evaluation, we observed that multi-horizon forecasting had better results than auto-regression,, which didn't translate to classification evaluation. In addition to that, the majority of regression predictions suffered from non-balanced precision-recall trade-off.

For the rest of the study, we decided to focus more on both multi-classification and regression of univariate time series at prediction horizon of 30 min. As regression might bring various benefits to the patient by providing glucose level fluctuations, while classification could provide better control on the models behavior. We decided to also further address the trustworthiness of trained models, mainly XGBoost, Transformer and Transformer-Cnn.

4.3.3 Data Augmentation

We evaluated models performance after applying data augmentation on the OhioT1DM dataset. 1366 hypoglycemia events labeled instances were added, and the chosen models were trained on classification tasks with prediction horizon of 30 mins.

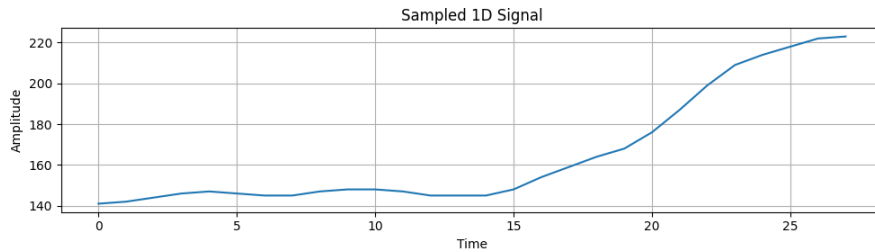


Figure 4.1: Example of synthetic CGM signal.

The added sequences were selected from a 100.000 instances synthetic dataset. The synthetic dataset was generated using a diffusion model trained on CGM dataset from the medium sized dataset (IOBP2), the generated signals (univariate time series of 28 consecutive CGM measure) were then clipped to ensure that values will always be in range [20 mg/l and 400 mg/l], as glycemic values shouldn't be out of this range based on observations.

Table 4.8: Evaluation of classification performance in the case of data augmentation.

Model	ACC	Hypo			Eu			Hype		
		Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
RF	0.932	0.802	0.765	0.783	0.953	0.942	0.947	0.905	0.932	0.919
XGB	0.934	0.802	0.766	0.783	0.958	0.941	0.949	0.904	0.943	0.923
LSTM	0.933	0.743	0.820	0.780	0.959	0.936	0.948	0.907	0.940	0.923
HYB	0.931	0.908	0.670	0.771	0.947	0.947	0.947	0.901	0.932	0.916
TRF	0.931	0.789	0.786	0.788	0.951	0.943	0.947	0.909	0.925	0.917
TRF-CNN	0.924	0.693	0.847	0.762	0.944	0.939	0.941	0.919	0.903	0.911
S4	0.922	0.802	0.768	0.785	0.919	0.965	0.941	0.945	0.852	0.896

Classification was not majorly affected by the addition of synthetic sequences. While most models showed an increase in recall at the expense of lower precision, TRF and HYB results showed the opposite behavior. Although the effect where minimal, it have lead to models with better balanced tendencies when it come to false negatives and false positives.

Table 4.9: Evaluation of regression performance in the case of data augmentation.

Model	MAE	RMSE	MAPE	R ²	PRC@30	REC@30	F1@30
RF	8.49	13.43	5.78	0.950	0.916	0.678	0.780
XGB	8.59	13.64	5.82	0.948	0.902	0.666	0.767
HYB	8.80	13.75	5.90	0.948	0.921	0.637	0.753
TRF	8.36	13.40	5.53	0.950	0.918	0.677	0.780
TRF-CNN	8.68	14.25	5.71	0.944	0.880	0.701	0.780
S4	8.67	13.27	6.00	0.951	0.953	0.599	0.736

In the case of regression, all models showed an improvement when evaluated on the regression task, we even had a new best performer (TRF with and RMSE of 13.4). The classification based on forecasting results didn't improve after the data augmentation, in some cases like the S4 it decreases.

The small impact introduced by data augmentation shows that in our use-case data augmentation might not have many upsides other than affecting the recall-precision trade-off and bringing a slight improvement regression performances.

4.3.4 Model Explainability

We used The SHAP library in python to calculate shap-values for the trained multiclass classification models. We averaged the results of the first 100 test instances, and plotted the shap-values related to the hypoglycemia event class.

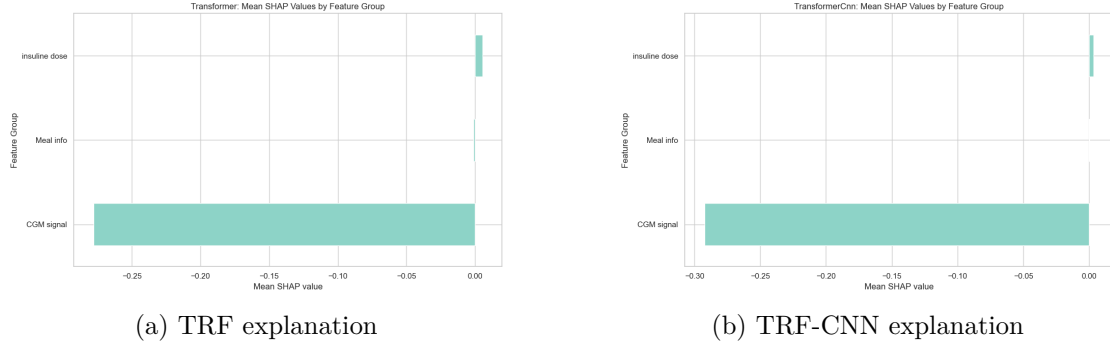


Figure 4.2: Explanation of trained models using SHAP.

From the plots in [Figure 4.2](#), it's clear that the prediction of the CGM signal is the main contributor to the prediction of the all models, while other features have low to null contribution.

Both Transformer and TransformerCnn show similar behaviour when it comes to feature importance, CGM signal is the main contributor to the prediction with a factor of -0.27 and -0.29 respectively. Insulin dose has a contribution of about 0.025 and the meal contribution was negligible. Based on the contribution sign, we can say that CGM signals tend to decrease the probability of detecting hypoglycemia , while the insulin dose tends to increase it. These observations are concord with the glycemic physiology.

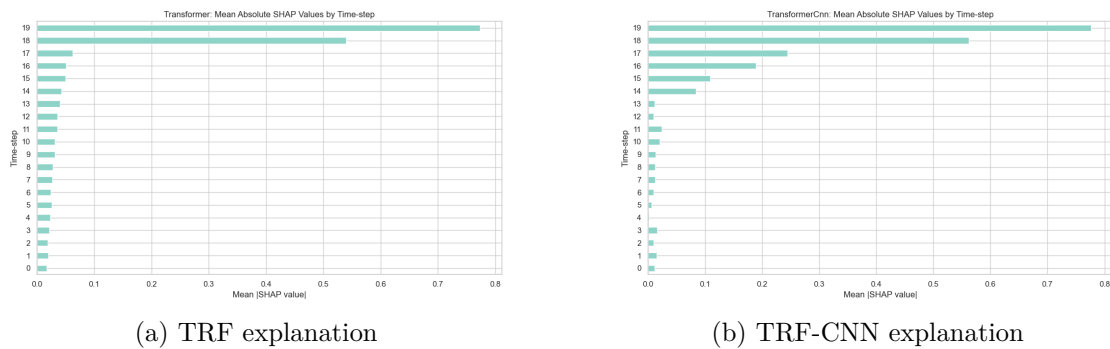


Figure 4.3: Explanation of trained models using SHAP.

If we move to the contribution of different time-steps in the prediction [Figure 4.3](#), we observe a trend where the most recent measures are the most important. The Transformer model seems to focus more on the two recent measures. While the TransformerCnn model, we observe that the main contributors are recent measures and the 4 oldest measures in the sequence.

4.3.5 Out of distribution evaluation:

Classification:

For the classification task, we compared the models based on the F1-score of the hypoglycemia class.

Table 4.10: Out-of-distribution Evaluation for Classification Task (F1-score)

Model	Train Test	Ohio	DCLP	IOPB
XGB	Ohio	0.779	0.508	0.534
	DCLP	0.425	0.606	0.601
	IOPB	0.448	0.625	0.615
LSTM	Ohio	0.786	0.503	0.534
	DCLP	0.402	0.643	0.603
	IOPB	0.447	0.653	0.626
Transformer	Ohio	0.770	0.472	0.497
	DCLP	0.491	0.646	0.627
	IOPB	0.351	0.594	0.579
TransformerCnn	Ohio	0.755	0.450	0.481
	DCLP	0.471	0.659	0.633
	IOPB	0.428	0.641	0.620
S4	Ohio	0.780	0.492	0.519
	DCLP	0.515	0.661	0.632
	IOPB	0.506	0.660	0.641

Regression

For the Regression task, we compared the models based on the Root-Mean-Squared-Error:

Table 4.11: Regression Out-of-Distribution evaluation (RMSE)(mg/dl)

Model	Train Test	Ohio	DLCP	IOBP
RF	Ohio	13.63	15.81	18.40
	DLCP	13.41	14.55	17.18
	IOBP	13.20	14.72	16.98
HYB	Ohio	13.81	15.81	18.52
	DLCP	13.27	13.81	17.03
	IOBP	12.91	14.27	16.73
Transformer	Ohio	13.32	15.09	17.54
	DLCP	13.23	13.75	16.91
	IOBP	12.78	14.15	16.60
TransformerCnn	Ohio	13.90	16.32	20.64
	DLCP	13.11	13.79	16.92
	IOBP	12.95	14.28	16.76
S4	Ohio	13.21	15.06	17.44
	DLCP	13.18	13.82	16.95
	IOBP	12.81	14.23	16.67

Classification from Regression:

For the classification task, we compared the models based on the precision, recall and F1-score of the hypoglycemia class.

Table 4.12: Out-of-Distribution evaluation of classification performance based on series forecasting

Model	Train Test	Ohio			DCLP			IOBP		
		Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
XGB	Ohio	0.903	0.658	0.761	0.863	0.571	0.687	0.837	0.555	0.667
	DCLP	0.912	0.661	0.766	0.891	0.552	0.682	0.869	0.540	0.666
	IOBP	0.939	0.637	0.759	0.906	0.536	0.674	0.890	0.524	0.659
HYB	Ohio	0.908	0.675	0.774	0.885	0.574	0.697	0.868	0.549	0.672
	DCLP	0.892	0.696	0.782	0.880	0.623	0.730	0.859	0.584	0.695
	IOBP	0.863	0.730	0.791	0.833	0.655	0.733	0.841	0.611	0.708
TRF	Ohio	0.908	0.683	0.780	0.87	0.589	0.703	0.855	0.567	0.682
	DCLP	0.885	0.712	0.789	0.859	0.649	0.740	0.848	0.600	0.703
	IOBP	0.927	0.680	0.789	0.896	0.587	0.710	0.890	0.551	0.681
TRF-CNN	Ohio	0.910	0.682	0.780	0.865	0.592	0.703	0.847	0.573	0.684
	DCLP	0.898	0.700	0.787	0.874	0.631	0.733	0.862	0.583	0.695
	IOBP	0.878	0.731	0.797	0.849	0.641	0.730	0.848	0.601	0.704
S4	Ohio	0.937	0.631	0.754	0.906	0.551	0.685	0.881	0.531	0.663
	DCLP	0.908	0.687	0.782	0.892	0.609	0.724	0.870	0.572	0.690
	IOBP	0.922	0.683	0.785	0.890	0.601	0.717	0.882	0.566	0.689

Observations

To see how models perform under different settings, we applied cross-dataset evaluation between three CGM datasets. We have taken the univariate approach and evaluated Classification, regression and classification based on regression. To our surprise, performance wasn't consistent between different datasets. It's clear that the performance on OhioT1DM is higher compared to the other two datasets, these results might be attributed to the differences in CGM devices and experimental protocols. It also means that patterns used to detect hypoglycemia are not shared between all three datasets. We also observed that the

The regression problem showed more consistent results for most models, and the performance translated well to different distributions. We also observed that the Ohio T1DM tended to have the lower RMSE regardless of the training dataset, followed closely by DCLP3 and finally the lowest score was recorded on the IOBP2 dataset. Even though evaluation on the IOBP2 dataset tends to have lower performance, the highest evaluation achieved on the OhioT1DM datasets was when training on the IOBP2 dataset.

When classifying the predicted measures, we have seen better F1-scores for hypoglycemia class than training on classification tasks directly, and the performance also translated well to other datasets different from the training one. We can assume that the learning performed on training datasets for the regression task holds out pretty well on different distributions. Still regression approach evaluation tend to have low recall.

The performance on the OhioT1DM dataset was consistently higher compared to the other two datasets. However, when models were trained on OhioT1DM and evaluated on different datasets, the results dropped significantly, highlighting a lack of generalization. In terms of regression performance for the 30-minute prediction horizon, our best result was an RMSE of 18.80 mg/dL, achieved by the multi-horizon Transformer trained on IOBP2 and evaluated on OhioT1DM. This compares favorably with previous studies: Mohebbi et al. [54] reported an RMSE of 20.08 mg/dL, while Zhu et al. [32] achieved 18.9 mg/dL. The best published performance we found was in [55], though their model was specifically trained for single-horizon forecasting at 30 minutes. In contrast, our models are trained to predict all six future glucose values simultaneously, offering a more comprehensive view of glycemic dynamics. When considering the full prediction horizon (i.e., all six steps), our models achieved an RMSE as low as 12.78 mg/dL, outperforming recent benchmarks such as [35], which reported an RMSE of 15.81 mg/dL under similar multi-horizon settings.

When comparing our results with previous studies that follow a classification-based approach, the highest F1-score achieved in our work was 0.797 at a prediction horizon of 30 minutes, using models trained on the OhioT1DM dataset. In comparison, similar work by Prendin et al. [28] reported an F1-score of 0.718. Although direct comparison is challenging due to differences in labeling strategies and dataset characteristics, these results serve as a useful benchmark and provide a degree of validation for the effectiveness of our approach.

4.3.6 Time Gain

In predictive monitoring, the “Prediction Horizon” is the length of time into the future, for which you are making predictions. Time Gain quantifies how much earlier a predictive model provides useful information about a future event or trend compared to simply observing the actual data as it unfolds. Essentially it measures the effective “warning time” your model gives you. We can distinguish three scenarios:

- **Ideal Scenario:** If a model perfectly predicts a glucose drop 30 minutes in advance ($PH = 30 \text{ min}$), and the prediction aligns perfectly in time with the actual future drop, the Time Gain is 30 minutes. The model has given you the full prediction horizon as warning time.
- **Delayed Prediction:** If the model predicts the drop, but its prediction of when the drop occurs is delayed (e.g., it predicts the drop to happen 10 minutes later than it actually does), the effective warning time is reduced. If PH is 30 min and the prediction is effectively “10 minutes late” in its timing of the event, the Time Gain might be $30 - 10 = 20$ minutes.
- **Premature Prediction:** If a model predicts an event too early, it might lead to premature action. The calculation of TG usually focuses on how much of the intended PH is realized as actual lead time for a correctly shaped prediction.

In applications like diabetes management, early warnings of hypoglycemia or hyperglycemia allow for timely intervention (e.g., consuming carbs, adjusting insulin). A model that predicts accurately but with no lead time (or worse, lags behind reality) is not clinically useful for proactive management. Moreover, It provides a more nuanced evaluation than just looking at point-wise accuracy (like RMSE). Two models might have similar RMSE, but one could offer a significantly better Time Gain, making it more valuable.

We used cross-correlation to determine when the predicted profile better aligned with the actual profile. We set the correlation mode to “full” as it is better for finding the optimal lag. If the lag is positive the predicted profile lags the actual profile, if it’s negative we say that the predicted profile leads the actual profile. We applied cross-dataset tests to get an idea on how time gain translates to out of distribution settings.

Table 4.13: Model performance comparison across different train-test combinations

Train	Test	OhioT1DM	DCLP3	IOLBP2
OhioT1DM	XGBoost	31.01±11.84	30.76±12.02	31.07±12.04
	S4	28.62±11.89	27.84±12.03	28.38±12.31
	Transformer	28.81±11.95	28.00±11.88	28.53±12.17
	TransformerCNN	28.69±11.89	27.98±12.02	28.52±12.33
DCLP3	XGBoost	30.27±12.03	29.87±11.87	30.03±12.03
	S4	28.38±11.51	27.76±11.63	28.78±12.01
	Transformer	28.81±11.99	27.93±11.62	28.59±12.08
	TransformerCNN	28.37±11.88	27.69±11.58	28.09±12.03
IOLBP2	XGBoost	30.64±12.04	30.25±11.97	30.43±11.98
	S4	28.19±11.61	27.86±11.60	28.24±12.89
	Transformer	29.06±11.75	28.37±11.78	28.72±12.03
	TransformerCNN	28.92±11.78	28.18±11.75	28.54±11.98

4.4 Discussions

This section interprets the findings presented in the previous section, in relation to the evaluation of trustworthiness of hypoglycemia prediction models based on CGM signals. The results of models explainability showed that information other than the CGM signals doesn’t contribute much to the models prediction. It could be said that, in the general case, there is no need to introduce basal insulin dosage and meal consumption to the predictive model. Thus, a univariate approach is preferred, as it limits the data shared by the user and ensures the ease of use by not requiring the repetitive task of introducing additional measures to the predictive pipeline.

Another potential question may arise concerning the class imbalance, and the use of data augmentation techniques to mitigate the lack of representation for hypoglycemia class. As have the results shown, the improvements marked by data augmentation using synthetic data generated by a diffusion model are really small . It mainly affects the precision-recall trade-off by increasing the recall and decreasing the precision. In our case, although the decrease in precision may sound problematic as it reduces the

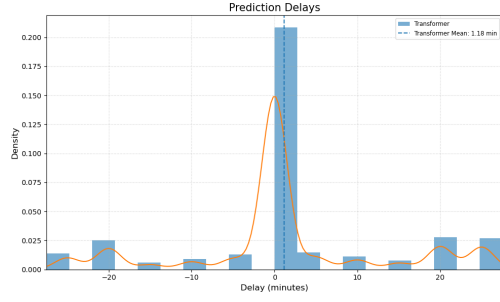
confidence of the users in the model’s predictions due to the high rate of false positives, providing a model with low recall (low true positives) may result in life threatening situations for the patients. Based on that, putting more emphasis on better recall scores is more important, and data augmentation is one way to achieve that.

To our surprise, the classification approach suffered the most from out-of-distribution evaluation and the introduction of the new datasets. The results previously achieved in the OhioT1DM datasets didn’t translate well to other datasets; we observed a decrease up to 0.2 in the hypoglycemia F1-score., which means that the patterns learned in one dataset don’t apply to other situations. On the other hand, the evaluation of the datasets IOBP2 and DCLP3 are close. This might be explained by the fact that both of them use similar CGM devices and both of the studies are related to the use of artificial pancreas.

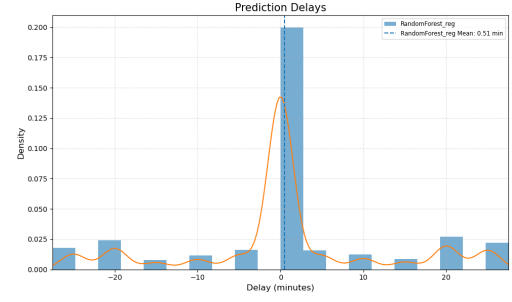
When it comes to the out-of-distribution evaluation of the regression task, results were more consistent across the board. We have seen some models performing better on a specific dataset when trained on a different dataset (best score on the test set of OhioT1DM [RMSE: 12.78] was achieved when trained on the IOBP2 dataset). Although OhioT1DM results are still more optimistic, the difference between the three datasets is considerably smaller, and the evaluations are closer. Even when we translate to the classification task from the regression, the performance still holds well, and the results are far superior to the case of direct classification for the IOBP2 and DCLP3 datasets (0.75 vs 0.7). Still, all classification results based on time series forecasting suffers from high precision and low recall.

Our main goal in this study is the prediction of hypoglycemia events to help the patients take preventive measures and decrease the risk of severe hypoglycemia events. The ability of the model to provide warnings early is measured through the Time Gain. From the evaluation, we found out that the XGBoost model had the best results when it comes to average time gain (negative delay, leads to early predictions). The main problem lies with the standard deviation of the delay, where all models delay had a standard deviation of around 12 minutes. This means that around 65% of the observed delays may exist in the range mean $TimeGain \pm stdmin$. Taking into consideration that the lag time of the cgm device may reach 15 mins, It would be problematic to have a large number of predictions reach a low time gain of 20 mins, to further investigate the distribution of the lag in the prediction, we plotted the histogram of the delays of different models.

As we can see for the case of XGBoost, the distribution of the delay resembles a normal



(a) Distribution of TRF delays.



(b) Distribution of XGB delays.

Figure 4.4: Histogram of forecasting delays.

distribution centered around a negative average of -0.98 mins, meaning that the predicted events tend to be detected in advance. A worrying observation is that the delays range from -25 mins to 25 mins, and extreme values occurred at an alarming rate (from the plot, the probability of $\text{abs}(\text{delay}) \geq 20$ mins is around 10%). A delay of 20 mins or more, regardless of the direction, could lead to wrong decision making that puts the patient's life in danger. To further evaluate the clinical accuracy of blood glucose predictions, and taking into consideration different ranges of glycemic adverse events, we use Clarke Error Grid.

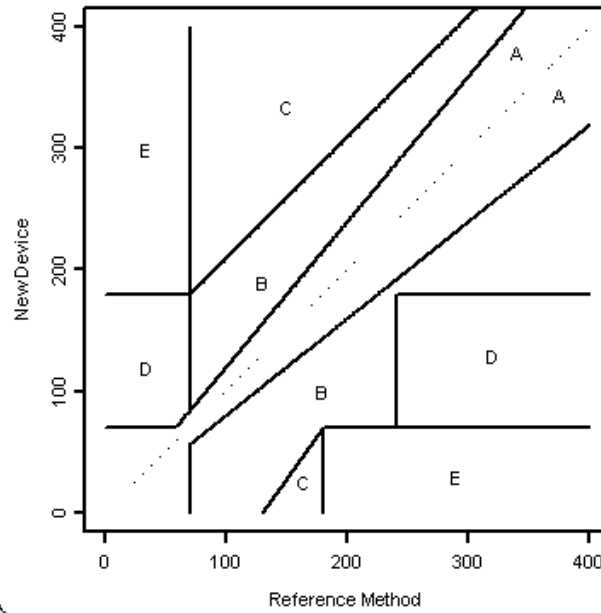


Figure 4.5: Clarke Error Grid.

The Clarke Error Grid [Figure 4.5](#) is a widely used method for evaluating the clinical accuracy of blood glucose predictions, particularly in diabetes management. It categorizes predicted glucose values against reference measurements into five zones (A to E) based on their potential clinical impact. Zone A includes predictions that are clinically accurate or result in no significant risk to the patient. Zone B covers predictions that deviate from the reference values but would lead to benign or no treatment errors. Zones C, D, and E represent increasingly dangerous inaccuracies—Zone C reflects unnecessary treatment, Zone D indicates failure to detect hypo- or hyperglycemia, and Zone E includes predictions that would lead to incorrect and potentially harmful treatment. The Clarke Grid provides a more clinically meaningful assessment than traditional statistical metrics like RMSE or R^2 , by focusing on the real-world consequences of prediction errors. It is especially relevant for continuous glucose monitoring (CGM) systems and hypoglycemia prediction models.

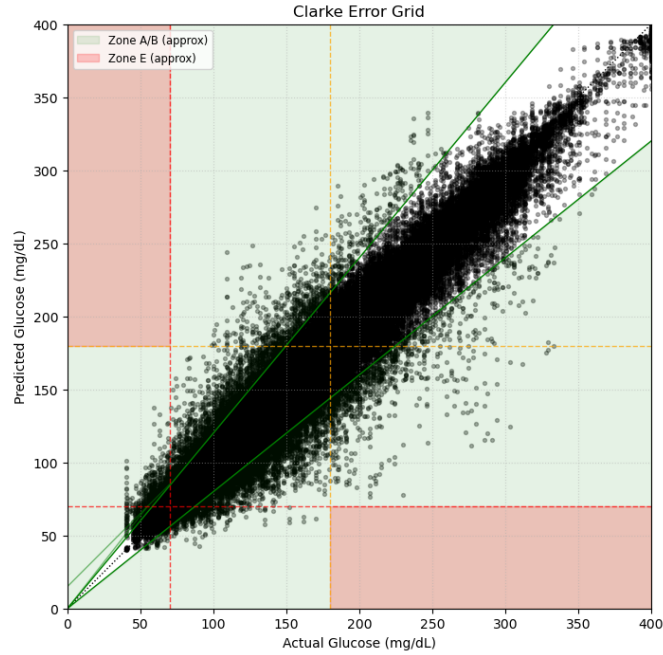


Figure 4.6: Clarke Error Grid for transformer.

We applied The Clarke error Grid to the models predictions, in order to get an idea about the clinical impact of the errors.

Table 4.14: Clinical evaluation of trained models

model	Zone A (%)	Zone B (%)	Zone C (%)	Zone D (%)	Zone E (%)
RF	94.41	4.39	0.002	1.20	0.0
XGB	94.38	4.31	0.005	1.29	0.0005
HYB	94.52	4.65	0.006	0.811	0.0
TRF	94.82	4.24	0.002	0.92	0.0
TRF-CNN	94.69	4.37	0.002	0.93	0.0
S4	94.59	4.29	0.00	1.10	0.0

All of the tested models had more than 98% of their predictions in the clinically safe zones of A and B, all the models had no predictions in the E range(except for XGB). The best performer seems to be HYB model as it has the lowest rate of predictions in Zone D.

Most of the prediction errors are present in the zone D, which leads to an error in the detection of hypoglycemia or hyperglycemia events. These results endorse the previous evaluation of classification task, taking into consideration that hypoglycemia is a minority label.

Another important axis to study is the trustworthiness of the prediction model. Fairness and privacy were ensured by not including any personal or private information unrelated to the prediction. We studied the explainability of the model to understand how different inputs and features contribute to the final prediction.

Conclusions

In this section, we evaluated the trustworthiness and generalizability of CGM-based hypoglycemia prediction models. Explainability analyses showed that CGM signals alone are sufficient for accurate predictions, supporting a univariate approach that enhances privacy and ease of use. While data augmentation slightly improved recall, it also reduced precision, reflecting the trade-off between sensitivity and false alarms—where higher recall is clinically preferable. Classification models struggled with generalization across datasets, especially from OhioT1DM to others, whereas regression models demonstrated more consistent out-of-distribution performance. Time gain analysis revealed that early predictions are achievable (e.g., by XGBoost), but high variability in delays poses safety risks. The Clarke Error Grid confirmed that most predictions fall in clinically acceptable

zones, with the HYB model minimizing dangerous misclassifications. Overall, CGM-only models can offer practical, private, and reliable hypoglycemia warnings, though challenges with delay variability and class imbalance remain.

Conclusion

In this study, we tried to build a trustworthy prediction model for hypoglycemia events that targets type 1 diabetes patients, all by taking into consideration out-of-distribution performance and consistency.

Based on the observed results, we found that the univariate approach offers similar performance to the multivariate one. We also concluded that the regression models generalizes better than the classification models on the task of hypoglycemia prediction.

We tried to evaluate the time gain and clinical accuracy of the regression model. Most of the tested models had an average time gain of 30 mins with a standard deviation of 11 mins. Which lead to having more than 90% of the predictions up to 20 mins of difference (delay or advance). This difference may result in unadapted treatment for the type 1 diabetes patient. To further investigate the clinical consequences of wrong predictions, more than 98% predictions of all models fall in the A and B zone of the Clarke error grid, which doesn't result in treatment error.

A possible limitation in our realized benchmarks is the lack of case specific hyper parameter tuning for the trained models and statistical significance testing of the obtained results (mainly due to computational resources limitations). We need also to mention that more datasets should have been included in the out of distribution evaluation. Another thing to consider is to further investigate how the inclusion of additional channels affect the performance in different scenarios.

Future Work

For future work, we aim to further implement new architectures, custom loss functions & preprocessing pipelines. In addition to study how the prediction errors vary based on the true values and ranges of glycemia.

We also aim to continue the development of the full suit of the prediction models for diabetes patients to help them in their daily decision taking. From adverse-events prediction to preventive action guidelines and healthy lifestyle advices. The developed suit could be deployed on phone devices to ensure availability and ease of use. One possible prospect is the prediction of how the consumption of glucose affects the glycemic levels, or how the injection of insulin affects the glycemic levels.

On another note, we want to venture into the digital-twins field by trying to make a digital twin of the diabetes patient in order to simulate how different factors affect the glycemic values.

Bibliographie

- [1] M. Z. BANDAY, A. S. SAMEER, and S. NISSAR, “Pathophysiology of diabetes: An overview,” *Avicenna journal of medicine*, vol. 10, no. 04, pp. 174–188, 2020.
- [2] N. TALEB, V. GINGRAS, R. CHENG, V. PARENT, V. MESSIER, D. BOVAN, A. SHOHOUDI, A.-S. BRAZEAU, and R. RABASA-LHORET, “Non-severe hypoglycemia in type 1 diabetes: A randomized crossover trial comparing two quantities of oral carbohydrates at different insulin-induced hypoglycemia ranges,” *Frontiers in Endocrinology*, vol. 14, p. 1186680, 2023. DOI: [10 . 3389 / fendo . 2023 . 1186680](https://doi.org/10.3389/fendo.2023.1186680). [Online]. Available: <https://pubmed.ncbi.nlm.nih.gov/37334295/>.
- [3] D. P. ZAHARIEVA, K. TURKSOY, S. M. MCGAUGH, R. POONI, T. VIENNEAU, T. LY, and M. C. RIDDELL, “Lag time remains with newer real-time continuous glucose monitoring technology during aerobic exercise in adults living with type 1 diabetes,” *Diabetes technology & therapeutics*, vol. 21, no. 6, pp. 313–321, 2019.
- [4] H. SHEIKH, C. PRINS, and E. SCHRIJVERS, “Artificial intelligence: Definition and background,” in *Mission AI: The new system technology*, Springer, 2023, pp. 15–41.
- [5] A. GÉRON, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* ” O’Reilly Media, Inc.”, 2022.
- [6] D. P. KINGMA and J. BA, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [7] B. LIM and S. ZOHREN, “Time-series forecasting with deep learning: A survey,” *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2194, p. 20200209, 2021.
- [8] S. HOCHREITER and J. SCHMIDHUBER, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

- [9] J. CHUNG, C. GULCEHRE, K. CHO, and Y. BENGIO, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv preprint arXiv:1412.3555*, 2014.
- [10] L. BOURDOIS, “Introduction to state space models (ssm),” *Hugging Face*, Jul. 2024, Retrieved May 9, 2025. [Online]. Available: <https://huggingface.co/blog/lbourdois/get-on-the-ssm-train>.
- [11] J. KIM, H. KIM, H. KIM, D. LEE, and S. YOON, “A comprehensive survey of deep learning for time series forecasting: Architectural diversity and open challenges,” *Artificial Intelligence Review*, vol. 58, no. 7, pp. 1–95, 2025.
- [12] C. MOLNAR, *Interpretable Machine Learning, A Guide for Making Black Box Models Explainable*, 3rd ed. 2025, ISBN: 978-3-911578-03-5. [Online]. Available: <https://christophm.github.io/interpretable-ml-book>.
- [13] M. T. RIBEIRO, S. SINGH, and C. GUESTRIN, “” why should i trust you?” explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.
- [14] S. M. LUNDBERG and S.-I. LEE, “A unified approach to interpreting model predictions,” *Advances in neural information processing systems*, vol. 30, 2017.
- [15] C. DUCKWORTH, M. J. GUY, A. KUMARAN, A. A. O’KANE, A. AYOBI, A. CHAPMAN, P. MARSHALL, and M. BONIFACE, “Explainable machine learning for real-time hypoglycemia and hyperglycemia prediction and personalized control recommendations,” *Journal of Diabetes Science and Technology*, vol. 18, no. 1, pp. 113–123, 2024.
- [16] S. WEISS, C. A. KULIKOWSKI, and A. SAFIR, “Glaucoma consultation by computer,” *Computers in Biology and Medicine*, vol. 8, no. 1, pp. 25–40, 1978.
- [17] J. BAJWA, U. MUNIR, A. NORI, and B. WILLIAMS, “Artificial intelligence in healthcare: Transforming the practice of medicine,” *Future healthcare journal*, vol. 8, no. 2, e188–e194, 2021.
- [18] R. YAMASHITA, M. NISHIO, R. K. G. DO, and K. TOGASHI, “Convolutional neural networks: An overview and application in radiology,” *Insights into imaging*, vol. 9, pp. 611–629, 2018.
- [19] J. JUMPER, R. EVANS, A. PRITZEL, T. GREEN, M. FIGURNOV, O. RONNEBERGER, K. TUNYASUVUNAKOOL, R. BATES, A. ŽÍDEK, A. POTAPENKO, *et al.*, “Highly accurate protein structure prediction with alphafold,” *nature*, vol. 596, no. 7873, pp. 583–589, 2021.

- [20] P. LEE, S. BUBECK, and J. PETRO, “Benefits, limits, and risks of gpt-4 as an ai chatbot for medicine,” *New England Journal of Medicine*, vol. 388, no. 13, pp. 1233–1239, 2023.
- [21] B. BEAULIEU-JONES *et al.*, *Machine learning for patient risk stratification: Standing on, or looking over, the shoulders of clinicians?* *npj digit med* 4 (1): 1–6, 2021.
- [22] M. P. KANTIPUDI, C. J. MOSES, R. ALUVALU, and S. KUMAR, “Remote patient monitoring using iot, cloud computing and ai,” *Hybrid Artificial Intelligence and IoT in Healthcare*, pp. 51–74, 2021.
- [23] N. AAGAARD, E. K. AASVANG, and C. S. MEYHOFF, “Discrepancies between promised and actual ai capabilities in the continuous vital sign monitoring of in-hospital patients: A review of the current evidence,” *Sensors (Basel, Switzerland)*, vol. 24, no. 19, p. 6497, 2024.
- [24] M. KIM, H. SOHN, S. CHOI, and S. KIM, “Requirements for trustworthy artificial intelligence and its application in healthcare,” *Healthcare Informatics Research*, vol. 29, no. 4, pp. 315–322, 2023.
- [25] D. DAVE, D. J. DESALVO, B. HARIDAS, S. MCKAY, A. SHENOY, C. J. KOH, M. LAWLEY, and M. ERRAGUNTLA, “Feature-based machine learning model for real-time hypoglycemia prediction,” *Journal of Diabetes Science and Technology*, vol. 15, no. 4, pp. 842–855, 2021.
- [26] A. MARCHETTI, L. BACCO, and S. PIERALICE, “Layered meta-learning algorithm for predicting adverse events in type 1 diabetes,”
- [27] M. CHENG, X. DIAO, Z. ZHOU, Y. CUI, W. LIU, and S. CHENG, *Toward short-term glucose prediction solely based on cgm time series*, 2024. arXiv: [2404.11924](https://arxiv.org/abs/2404.11924) [cs.AI]. [Online]. Available: <https://arxiv.org/abs/2404.11924>.
- [28] F. PRENDIN, S. DEL FAVERO, M. VETTORETTI, G. SPARACINO, and A. FACCHINETTI, “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*, vol. 21, no. 5, p. 1647, 2021.
- [29] A. ZHANG, Z. C. LIPTON, M. LI, and A. J. SMOLA, *Dive into Deep Learning*. Cambridge University Press, 2023, <https://D2L.ai>.
- [30] F. PRENDIN, J. PAVAN, G. CAPPON, S. DEL FAVERO, G. SPARACINO, and A. FACCHINETTI, “The importance of interpreting machine learning models for blood glucose prediction in diabetes: An analysis using shap,” *Scientific reports*, vol. 13, no. 1, p. 16 865, 2023.

- [31] G. CAPPON, M. VETTORETTI, G. SPARACINO, S. DEL FAVERO, and A. FACCHINETTI, “Replaybg: A digital twin-based methodology to identify a personalized model from type 1 diabetes data and simulate glucose concentrations to assess alternative therapies,” *IEEE Transactions on Biomedical Engineering*, vol. 70, no. 11, pp. 3227–3238, 2023.
- [32] T. ZHU, I. AFENTAKIS, K. LI, R. ARMIGER, N. HILL, N. OLIVER, and P. GEORGIOU, “Multi-horizon glucose prediction across populations with deep domain generalization,” *IEEE Journal of Biomedical and Health Informatics*, 2024, Accepted author’s manuscript. DOI: 10.1109/JBHI.2024.3428921. [Online]. Available: <https://discovery.ucl.ac.uk/id/eprint/10197097/>.
- [33] R. SERGAZINOV, M. ARMANDPOUR, and I. GAYNANOVA, “Gluformer: Transformer-based personalized glucose forecasting with uncertainty quantification,” in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2023, pp. 1–5.
- [34] S.-M. LEE, D.-Y. KIM, and J. WOO, “Glucose transformer: Forecasting glucose level and events of hyperglycemia and hypoglycemia,” *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 3, pp. 1600–1611, 2023. DOI: [10.1109/JBHI.2023.3236822](https://doi.org/10.1109/JBHI.2023.3236822).
- [35] M. A. KARAGOZ, M. D. BRETON, and A. EL FATHI, *A comparative study of transformer-based models for multi-horizon blood glucose prediction*, Presented at the IFAC Diabetes Technology Conference, Valencia, Spain, May 2025, 2025. arXiv: [2505.08821 \[q-bio.QM\]](https://arxiv.org/abs/2505.08821).
- [36] B. KOVATCHEV and W. CLARKE, *Peculiarities of the continuous glucose monitoring data stream and their impact on developing closed-loop control technology*, 2008.
- [37] N. V. CHAWLA, K. W. BOWYER, L. O. HALL, and W. P. KEGELMEYER, “Smote: Synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.
- [38] D. P. KINGMA and M. WELLING, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
- [39] I. GOODFELLOW, J. POUGET-ABADIE, M. MIRZA, B. XU, D. WARDE-FARLEY, S. OZAIR, A. COURVILLE, and Y. BENGIO, “Generative adversarial networks,” *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.
- [40] P. DHARIWAL and A. NICHOL, “Diffusion models beat gans on image synthesis,” *Advances in neural information processing systems*, vol. 34, pp. 8780–8794, 2021.

- [41] S. L. CICHOSZ and A. A. P. XYLANDER, “A conditional generative adversarial network for synthesis of continuous glucose monitoring signals,” *Journal of Diabetes Science and Technology*, vol. 16, no. 5, pp. 1220–1223, 2022.
- [42] O. RONNEBERGER, P. FISCHER, and T. BROX, “U-net: Convolutional networks for biomedical image segmentation,” *arXiv preprint arXiv:1505.04597*, 2015.
- [43] L. BREIMAN, “Random forests,” *Machine Learning*, vol. 45, pp. 5–32, 2001.
- [44] T. CHEN and C. GUESTRIN, “Xgboost: A scalable tree boosting system,” in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.
- [45] C. SZEGEDY, W. LIU, Y. JIA, P. SERMANET, S. REED, D. ANGUELOV, D. ERHAN, V. VANHOUCHE, and A. RABINOVICH, “Going deeper with convolutions,” *arXiv preprint arXiv:1409.4842*, 2014.
- [46] A. VASWANI, N. SHAZEER, N. PARMAR, J. USZKOREIT, L. JONES, A. N. GOMEZ, L. KAISER, and I. POLOSUKHIN, “Attention is all you need,” *arXiv preprint arXiv:1706.03762*, 2017.
- [47] A. RADFORD, K. NARASIMHAN, T. SALIMANS, and I. SUTSKEVER, “Improving language understanding by generative pre-training,” *arXiv preprint arXiv:1805.00572*, 2018.
- [48] A. GU, I. JOHNSON, K. GOEL, K. SAAB, T. DAO, A. RUDRA, and C. RÉ, *Combining recurrent, convolutional, and continuous-time models with linear state-space layers*, 2021. DOI: [10.48550/arXiv.2110.13985](https://doi.org/10.48550/arXiv.2110.13985). arXiv: [2110.13985](https://arxiv.org/abs/2110.13985) [cs.LG]. [Online]. Available: <https://arxiv.org/abs/2110.13985>.
- [49] A. GU, K. GOEL, and C. RÉ, “Efficiently modeling long sequences with structured state spaces,” *arXiv preprint arXiv:2111.00396*, 2021.
- [50] A. GU, T. DAO, S. ERMON, A. RUDRA, and C. RÉ, “Hippo: Recurrent memory with optimal polynomial projections,” *Advances in neural information processing systems*, vol. 33, pp. 1474–1487, 2020.
- [51] C. MARLING and R. BUNESCU, “The ohiot1dm dataset for blood glucose level prediction: Update 2020,” in *CEUR workshop proceedings*, vol. 2675, 2020, p. 71.
- [52] Z. LI, “Iobp pivotal trial statistical analysis plan: Bp aspart/lispro vs. standard care,” Bionic Pancreas Study, Tech. Rep., version 4.0, 2021. [Online]. Available: https://cdn.clinicaltrials.gov/large-docs/13/NCT04200313/SAP_011.pdf.
- [53] N. I. of DIABETES, DIGESTIVE, and K. DISEASES, *The international diabetes closed loop (idcl) trial: Clinical acceptance of the artificial pancreas (dclp3)*, Accessed:

- 2025-06-13, 2025. [Online]. Available: <https://repository.niddk.nih.gov/study/292>.
- [54] A. MOHEBBI, A. R. JOHANSEN, N. HANSEN, P. E. CHRISTENSEN, J. M. TARP, M. L. JENSEN, H. BENGTSSON, and M. MØRUP, *Short term blood glucose prediction based on continuous glucose monitoring data*, arXiv preprint arXiv:2002.02805, version 2, 2020. arXiv: [2002.02805](https://arxiv.org/abs/2002.02805) [eess.SP].
 - [55] S. GHIMIRE, T. CELIK, M. GERDES, and C. W. OMLIN, “Deep learning for blood glucose level prediction: How well do models generalize across different data sets?” *PLOS ONE*, vol. 19, no. 9, e0310801, 2024. DOI: [10.1371/journal.pone.0310801](https://doi.org/10.1371/journal.pone.0310801). [Online]. Available: <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0310801>.