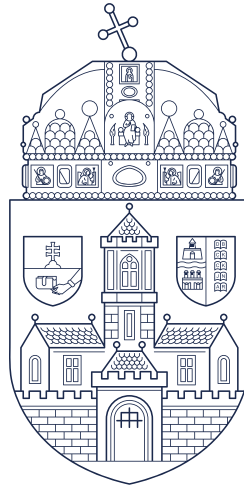


Óbuda University

PhD Thesis



Development of machine learning solutions for modeling and
automation of processes

by

Lehel Dénes-Fazakas

Supervisors:

Dr. habil György Eigner

Prof. Dr. habil László Szilágyi

Doctoral School of Applied Informatics and Applied Mathematics
Budapest

2025

Abstract

This dissertation presents novel machine learning solutions for modeling and automating processes in healthcare, with a primary focus on diabetes management and medical image processing. In the diabetes domain, the work introduces methods to detect patients physical activity from both synthetic and real-world data, integrating continuous glucose monitoring (CGM) and heart rate signals. By employing conservative machine learning algorithms alongside more advanced recurrent neural networks (RNNs), the dissertation demonstrates how accurate exercise detection can be achieved even under real-life conditions thereby enhancing personalized feedback for patients.

Beyond activity detection, the research also addresses the automated recognition of food intake gestures using wearable sensor data. This solution is crucial for improving dietary records in diabetes therapy, as it accurately identifies when carbohydrate consumption occurs. The dissertation further proposes a reinforcement learning (RL) framework for insulin regulation, wherein policy and value networks learn optimal dosing strategies based solely on CGM data. Experiments show that this RL-based controller robustly maintains blood glucose within safe limits, offering a foundation for more adaptive and patient-specific “artificial pancreas” systems.

In medical image processing, the dissertation I developed advanced architectures for segmenting and classifying magnetic resonance imaging (MRI) scans. Specifically, it introduces refined U-net designs ranging from 2D to spatiotemporal 3D to segment infant brain tissues into cerebrospinal fluid, gray matter, and white matter with high accuracy. Furthermore, a specialized “L-net” approach combines a U-net segmentation backbone with a convolutional neural network (CNN) classifier to distinguish glioma, meningioma, and pituitary tumors. Extensive experimentation reveals state-of-the-art performance in MRI tissue segmentation and tumor classification, underscoring the potential of deep learning to enhance diagnostic precision.

Collectively, these contributions highlight how machine learning-driven methods can automate critical aspects of clinical management and medical imaging. By uniting algorithmic innovations with real-world application needs, the dissertation underscores the transformative potential of data-driven approaches in modern healthcare.

Kivonat

Ez a disszertáció újszerű gépi tanulási megoldásokat mutat be az egészségügyben zajló folyamatok modellezésére és automatizálására, elsősorban a cukorbetegség kezelésére és az orvosi képfeldolgozásra összpontosítva. A diabétesz területén a munka olyan módszereket mutat be, amelyekkel szintetikus és valós adatokból egyaránt detektálható a betegek fizikai aktivitása, integrálva a folyamatos glükózmonitorozás (CGM) és a pulzusszám jeleit. A disszertáció egyszerű gépi tanulási algoritmusok alkalmazásával a fejlettebb rekurrens neurális hálózatok (RNN) mellett bemutatja, hogyan érhető el pontos mozgásérzékelés akár valós körülmények között is, ezáltal javítva a betegek személyre szabott terápiáját.

Az aktivitásérzékelésen túl a kutatás foglalkozik az étkezési gesztusok automatizált felismerésével is, viselhető szenzoradatok felhasználásával. Ez a megoldás kulcsfontosságú a diabéteszterápia diétás nyilvántartásának javításához, mivel pontosan azonosítja, mikor történik szénhidrátfogyasztás. A disszertáció továbbá egy megerősítési tanulási (RL) keretrendszerrel javasol az inzulinszabályozáshoz, amelyben a politika és értékhálózatok kizárólag a CGM-adatok alapján tanulják meg az optimális adagolási stratégiákat. A kísérletek azt mutatják, hogy ez az RL-alapú szabályozó robusztusan tartja a vércukorszintet biztonságos határokon belül, ami alapot kínál az adaptívabb és páciens-specifikusabb „mesterséges hasnyálmirigy” rendszerekhez.

Az orvosi képfeldolgozás területén a disszertáció fejlett architektúrákat fejleszt a mágneses rezonancia képalkotó (MRI) felvételek szegmentálására és osztályozására. Konkrétabban, a 2D-től a térbeli 3D-ig terjedő, kifinomult U-háló terveket mutat be, hogy a csecsemők agyszöveteit nagy pontossággal szegmentálja agy-gerincvelői folyadékra, szürkeállományra és fehérállományra. Továbbá egy speciális „L-háló” megközelítés az U-háló szegmentációs gerincet egy konvolúciós neurális hálózat (CNN) osztályozóval kombinálja a glióma, a meningeóma és az agyalapi mirigy daganatok megkülönböztetésére. A kiterjedt kísérletezés az MRI-szövetek szegmentálása és a tumorok osztályozása terén a legmodernebb teljesítményt mutatja, kiemelve a mélytanulásban rejlő lehetőségeket a diagnosztikai pontosság fokozására.

Ezek a hozzájárulások együttesen rávilágítanak arra, hogy a gépi tanulás által vezérelt módszerek hogyan automatizálhatják a klinikai kezelés és az orvosi képalkotás kritikus aspektusait. Az algoritmikus innovációk és a valós alkalmazási igények egyesítésével a disszertáció kiemeli az adatvezérelt megközelítések átalakító potenciálját a modern egészségügyben.

Acknowledgements

I would like to thank Dr. Prof. habil László Szilágyi and Dr. György Eigner for their work. For the time they dedicated to my doctoral research work, during which I had the opportunity to work on several types of medical research. They gave me the opportunity to do research in the field of diabetes and medical image processing. In this way, I had the opportunity not only to deepen my knowledge of artificial intelligence, but also to broaden my knowledge of biology. I would like to thank the Centre for Life Science Regulation Research at the University of Óbuda's Centre for University Research and Innovation (CERI) for hosting me and providing me with the space and space to conduct my PhD research. I would like to thank the staff of the Doctoral School of Applied Informatics and Applied Mathematics at the University of Óbuda for helping me with any problems I encountered. They also supported me in attending the conference. Finally, I would like to thank COMPUTER ENGINEERING AND AUTOMATICS RESEARCH INSTITUTIONS and WIGNER PHYSICAL RESEARCH CENTRE for providing me with virtual machines through A HUN-REN Cloud (formerly ELKH Cloud). This made it easier to run algorithms using large computational capacity. Also, the flow of courses over long periods of time.

Contents

List of Figures	8
List of Tables	10
List of abbreviations	13
1 Introduction	15
1.1 Diabetes	16
1.2 Magnetic resonance imaging	18
1.2.1 Advantages of MRI	20
1.2.2 Disadvantages of MRI	20
2 Performance Evaluation Metrics	22
2.1 Area Under the Curve (AUC) and ROC Curve	22
2.2 Confusion Matrix	23
2.3 Derived Metrics: Recall, Precision, Accuracy, FPR, F1 Score, Specificity	24
2.4 Boxplot	25
2.5 Control Variability Grid Analysis (CVGA)	26
2.6 Time in Range (TIR)	27
3 Diabetes-related research	29
3.1 Chapter Overview	31
3.2 Novel Contributions	31
3.3 Physical activity detection	32
3.3.1 Goal	32
3.3.2 Introduction	32
3.3.3 Synthetic Dataset	34
3.3.4 Tested models on synthetic dataset	36
3.3.5 Results on synthetic dataset	36
3.3.6 Real Patient Datasets	39
3.3.7 Explored Machine Learning Techniques	43
3.3.8 Test Cases	47
3.3.9 Feature extraction	48
3.3.10 Results Real Patient Dataset	50
3.3.11 OHIO T1DM Dataset for Recurrent Neural network	51
3.3.12 Recurrent neural network approach	53
3.3.13 Result with Recurrenct Neural Network	57
3.3.14 Conclusion	63
3.4 Gesture detection	64
3.4.1 Goal	64
3.4.2 Introduction	64
3.4.3 Data collection	65
3.4.4 Machine learning based approach	67

3.4.5	Result with machine learning approach	67
3.4.6	Clemson dataset	68
3.4.7	LSTM based method	75
3.4.8	Training testing	76
3.4.9	Validation	77
3.4.10	Results with LSTM model	77
3.4.11	Conclusion	79
3.5	Insulin control with reinforcement-based neural network	80
3.5.1	Goal	80
3.5.2	Introduction	80
3.5.3	Virtual Patient model	82
3.5.4	Reinforcement Learning	83
3.5.5	Closed Loop	84
3.5.6	Meal generator	85
3.5.7	First results	86
3.5.8	Second results	88
3.5.9	Third Results	89
3.5.10	Final neural network	90
3.5.11	Training Phase	92
3.5.12	Testing Phase	93
3.5.13	Result and Discussion	93
3.5.14	Conclusion	96
3.6	Thesis Summary – Diabetes Research	97
4	Processing MRI images	99
4.1	Chapter Overview	100
4.2	Novel Contributions	100
4.3	Infant brain tissue segmentation	101
4.3.1	Goal	101
4.3.2	Introduction	101
4.3.3	Dataset and Challenge Context	102
4.3.4	Data Preprocessing and Intensity Alignment	102
4.3.5	First U-net network	104
4.3.6	First result	106
4.3.7	Second U-net network	106
4.3.8	Second results	108
4.3.9	Spatiotemporal Convolutions: Unleashing Dynamics in Deep Learning . .	109
4.3.10	Third U-net	110
4.3.11	Third results	113
4.3.12	Conclusion	118
4.4	Tumor classification of MRI scans	118
4.4.1	Goal	119
4.4.2	Introduction	119
4.4.3	Dataset	120
4.4.4	First Network architectures	120
4.4.5	First results	121
4.4.6	L-net (U-net plus CNN), Second Architectures	124
4.4.7	Training and testing	128
4.4.8	L-net Results	128
4.4.9	Conclusion	133
4.5	Thesis Summary – MRI Research	133

5	Conclusion	135
6	Future Research Directions	136
6.1	Detection of Physical Activity	136
6.2	Gesture Detection	136
6.3	Reinforcement Learning-Based Insulin Regulation	136
6.4	Infant Brain Tissue Segmentation	137
6.5	Brain Tumor Detection	137
7	Contribution in publications	138
8	Appendix	139
8.1	Physical activity supplementary	139
8.1.1	Precision	139
8.1.2	Recall	144
8.1.3	Accuracy	149
8.1.4	Analyzation of the best 30 models	153
8.2	Reinforcement learning based insulin control supplementary	156
	Bibliography	158
	Own Publications Pertaining to Theses	174
	Publications not Pertaining to Theses	176

List of Figures

1.1	Artificial Pancreas [R11]	17
1.2	MRI machine [R12]	18
1.3	MRI captures types [R17]	19
2.1	AUC metric [R22]	23
2.2	Confusion Matrix [R24]	23
2.3	Boxplot [R26]	26
2.4	CVGA diagram and zone description	27
2.5	Time in range plots [R29]	28
3.1	Jacobs simulator model output indicating the physical activity with the red section on figure, without CGMS (top) and with CGMS (bottom)	35
3.2	ROC curve of models	38
3.3	AUC values obtained by various classification models	39
3.4	The ways of data extraction from the Ohio T1DM Dataset at a given patient – graphical example. The black arrows represent a 24 hours long block according to the time stamps, midnight to midnight. In this example, the blue regions belong to those 24 hours long blocks, where CGMS data were available and exercise was reported without available corresponding HR data. The green region represents a 24 hours long data block where CGMS data were available, exercise was reported and the corresponding HR data were available. The transparent area covered by orange dashed line represents self reported exercise (24 hours long block during which the exercise happened) without corresponding CGMS signal. The transparent area covered by red dashed line represents a 24 hours long block during which probably an exercise event happened but it was not reported (“non annotated outcome”).	40
3.5	Abstraction of feature extraction during operation. The v , vp , vpp and ap features are originated from the d kind features, thus these are not listed here. In the case of the dp_i only the first four value has represented as a demonstration, however, all sampled values has been considered from the window during operation.	50
3.6	AUC of models in all use-cases – test results	51
3.7	The structure of the used GRU and LSTM models. All model configurations had the same structure, only the values of the hyper parameters changed. The kernel is represented in the image using a matrix of how much data it processes.	56
3.8	F1 score value for different number of RNN cell	57
3.9	F1 score value for different number of Look back	58
3.10	F1 score value for different datatype	59
3.11	F1score value for different number of Drop out rate	60
3.12	F1 score value for different number of Dense neurons	61
3.13	Food Consumption Detection system in plan, A: Hand gesture, B: Raw sensor data, C: Smart Watch/Band, D: Collected-Transformed data, E: Deep learning model, F: No Eating, G: Eating	65

3.14	Clemson All-day dataset data collection road map A: Patient Population, B: Raw data collection, C: ActiGraph gt9x, D: Shimmer 3, E: Export Data, F: Self reported eating, G: Exported data	68
3.15	Food consumption txt example	69
3.16	Directory structure	69
3.17	CSV file example	70
3.18	My proposed method in real life use, A: Hand gesture, B: Raw sensor data, C: Smart Watch/Band, D: Collected data, E: Smart Phone, F: Transformed data, G: LSTM modell, H: No Eating, I: Eating, J: Export, K: Database	71
3.19	Data aggragation	72
3.20	Labelling	72
3.21	Transformed data for training	73
3.22	Sample patient accelerometer data after transformation	73
3.23	Sample patient gyroscope data after transformation	74
3.24	Sample patient magnetometer data after transformation	74
3.25	Sample patient quaternion data after transformation	75
3.26	LSTM model	76
3.27	Boxplot for metric	78
3.28	Confusion matrix	79
3.29	The top figure represents the block diagram of the reinforcement learning based closed-loop algorithm. The bottom figure details the control architecture of the system.	85
3.30	Investigated reward functions.	89
3.31	A graphical representation of the policy and value networks as depicted by Pytorch. The image on the left is the policy network, which is larger than the value network. This network learns the control strategy. On the right is the value network, which gives an estimate of how good the policy network was.	90
3.32	The top figure represents the architecture of the Policy Network, which consists of multiple Exponential Linear Unit (ELU) blocks followed by an Identity block and a final Linear activation layer. The middle figure shows the architecture of the Value Network, which is similarly structured with a sequence of ELU blocks, an Identity block, and a Linear activation layer at the output. The bottom figure illustrates the internal structure of the ELU and Identity blocks, where the ELU block consists of a linear layer followed by an ELU activation, and the Identity block replaces the ELU with an identity activation function after the linear layer.	91
3.33	Blood glucose median curve with standard deviations and meal intakes for all the investigated days.	94
3.34	Mean blood glucose curve with standard deviation and meal intakes for all the investigated days.	94
3.35	CVGA diagram for all tested days	95
4.1	A whole volumetric MRI record presented in cross-sections: T1 data in top panel, T2 data in the middle panel, ground truth in the bottom panel.	103
4.2	The U-net network structure deployed for brain tissue segmentation.	104
4.3	The structure of an encoder block, using Conv2D layers with kernel size 3×3 , ReLU activation function, and MaxPooling2D kernel size 2×2	105
4.4	The structure of the network's bridge part, using Conv2D kernel size 3×3 , and ReLU activation function.	105
4.5	The structure of a decoder block, using Conv2DTranspose kernel size 2×2 , Conv2D kernel size 3×3 , and ReLU activation function.	105
4.6	The proposed 2D U-net architecture.	107
4.7	The proposed 3D U-net architecture.	107

4.8	The structure of: (a) encoder block; (b) bridge part of network; (c) decoder block. Convolution works in 2D or 3D, depending on the architecture model.	108
4.9	2 plus 1 dimensional convolution	110
4.10	The proposed 3D U-net architecture.	111
4.11	Structure of an encoder block.	112
4.12	Structure of the bridge part.	112
4.13	Structure of a decoder block.	113
4.14	Boxplot of different segmentation benchmark metrics	115
4.15	Benchmark values obtained for individual records and tissue types in panels (a) to (c); accuracy rates obtained for individual records in panel (d).	117
4.16	All slices of a segmented brain. The three shades of green from dark to light represent the correctly segmented pixels of the three main tissue types: CSF, GM and WM, respectively, while red color indicates misclassified pixels.	118
4.17	The proposed network architectures: (a) with dropout; (b) without dropout. . .	121
4.18	An overview of the L-net architecture, combining segmentation and classification tasks. The model first segments the tumor region using a U-net structure (left), where the black image represents the segmentation mask as the output. It then classifies the tumor, based on the features extracted by the U-net, into one of three categories: Glioma, Meningioma, or Pituitary (right), using a CNN.	125
4.19	The U-net module, which forms a key part of the overall L-net architecture. . . .	126
4.20	Encoder blocks of the U-net part of the L-net.	126
4.21	The bridge section within the U-net component of the L-net architecture.	126
4.22	The decoder blocks within the U-net component of the L-net architecture.	127
4.23	The detailed architecture of the CNN component that forms a crucial part of the overall L-net structure.	127
4.24	The individual blocks of the CNN component within the broader L-net architecture.	128
4.25	A boxplot for the precision, recall, and F1 score metrics. Lower axis indicates the image size and metric as well.	129
4.26	Boxplot for AUC metrics obtained for different tumor types and various imagesizes.	130
4.27	Some examples of correct and mistaken decisions made by the L-net architecture at the image size 128×128 , organized in the format of a confusion matrix. The Pituitary class has no false positives.	132
8.1	Precision value for different number off RNN cell	139
8.2	Precision value for different number off Look back	140
8.3	Precision value for different datatype	141
8.4	Precision value for different number off Drop out rate	142
8.5	Precision value for different number off Dense neurons	143
8.6	Recall value for different number off RNN cell	144
8.7	Recall value for different number off Look back	145
8.8	Recall value for different datatype	146
8.9	Recall value for different number off Drop out rate	147
8.10	Recall value for different number off Dense neurons	148
8.11	ACC value for different number off RNN cell	149
8.12	ACC value for different number off looc back	150
8.13	ACC value for different datatype	151
8.14	ACC value for different number off Drop out rate	152
8.15	ACC value for different number off Dense neurons	153
8.16	Value network	156
8.17	Policy network	157

List of Tables

3.1	Normalized confusion matrix obtained by the use of various classification methods	37
3.2	Properties of the cleaned data from the Ohio T1DM dataset used for patients' blood glucose feature extraction.	42
3.3	Properties of the cleaned data from the Ohio T1DM dataset used for the extraction of patients' blood glucose and heart rate features.	42
3.4	Properties of the cleaned data from the D1namo dataset used for patients' blood glucose feature extraction.	44
3.5	Properties of the cleaned data from the D1namo dataset used for the extraction of patients' blood glucose and heart rate features.	44
3.6	AUC performance achieved in different tests	51
3.7	Glucose hours by patients	52
3.8	The 30 best model F1 scores	62
3.9	Accuracy results with logistic regression and MLP	67
3.10	Accuracy results with random forest	67
3.11	Description of metrics	78
3.12	Patient parameter sets used in the Identifiable Virtual Patient (IVP) model. Rows with identical values for VG and BW represent data from the same patient. The parameters include BW (Body Weight), GEZI (Gastric Emptying and Insulin Sensitivity), EGP (Endogenous Glucose Production), CI (Clearance Index), SI (Sensitivity Index), tau1 (Delay Parameter 1), tau2 (Delay Parameter 2), p2 (Parameter 2), and VG (Volume of Glucose). In the case where the parameters (BW) and (VG) are the same, I are talking about the same patient. But with night and day parameter sets.	82
3.13	Aggregated table for all investigated simulation days, including mean, maximum, minimum, standard deviation, and quarterlies values in terms of TIR zones. . . .	93
4.1	Average values of the main accuracy indicators	106
4.2	Average accuracy indicator values obtained from the ten MRI records	109
4.3	Recall, precision and F1 score for all patients, displayed in increasing order of the F1 score	114
4.4	Accuracy benchmark or rate of correct decisions, obtained for different patients, sorted in increasing order	114
4.5	The distribution of various metrics values	115
4.6	Statistics of the ten confusion matrices obtained for individual patients	116
4.7	Benchmark comparison with previous works and state-of-the-art methods	118
4.8	The structure of the image data set involved in the study	120
4.9	Average and standard deviation of the main accuracy indicator (F1 score) obtained by various network models. For each of the five runs, each performed using one group of input data for evaluation and the other four for training, the average of the three F1 scores (for classes Meningioma, Glioma, and Pituitary) were extracted. The average and the standard deviation of these five values are reported here for each network model.	122
4.10	Precision and recall values obtained by the L-net model.	129

4.11	AUC benchmarks obtained for different tumor classes	130
4.12	Overall confusion matrices obtained by the best four VGG models [C7] (upper row), and the proposed L-net architecture at various image resolutions.	131
8.1	The 30 best model AUC and ACC scores	154
8.2	The 30 best model Precision and Recall scores	155

List of abbreviations

Abbreviation	Meaning
IVP	Identifiable Virtual Patient
BG	Blood glucose
EGP	endogenous glucose production
CGMS	Continuous Glucose Monitoring System
CHO	Carbohydrate
MRI	Magnetic resonance imaging
AI	Artificial Intelligence
CNN	Convolutional neural network
CNNs	Convolutional neural networks
CSF	Cerebro-spinal fluid
GM	Grey matter
WM	White matter
ELU	Exponential Linear Unit
TPR	Sensitivity or True Positive Rate
TNR	Specificity or True Negative Rate
PPV	Precision or Positive Predictive Value
DSC	Dice Similarity Score
ACC	Accuracy
KNN	K-nearest neighbors
HR	Hearth rate
BiRNN	Bidirectional Recurrent Neural Networks
GRU	Gated Recurrent Unit
AUC	Area Under Curve
ROC	Receiver Operating Characteristic Curve
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
ADAM	Adaptive Moment Estimation
AI	Artificial Intelligence
ReLU	Rectified Linear Unit
TP	True Positive
IMU	Inertial Measurement Unit
ReLU	Rectified Linear Unit
GRU	Gated Recurrent Units
CAD	Clemson ALL-day dataset
P(E)	Probability of eati
M/s ²	Metres per syecond square
deg/sec	Degrees per second
uTesla	Micro Tesla
IIR	infinite impulse response

Hertz	Hz
Identity	ID
msecs	milliseconds
Gyro	Gyroscopoe
Accel	Accelerometer
Mag	Magnetometer
LR	Logistic regression
RFC	Random Forest
MLP	Multi layer preceptron
h_t	hidden state
t	time step
BPTT	backpropagation through time
MPU	Motion Processing Unit
MLP	Motion Processing Library
6DOF	6 degrees of freedom
Quat	Quaternion
DM	Diabetes mellitus
T1DM	Type 1 Diabetes Mellitus
T2DM	Type 2 Diabetes Mellitus
AP	Artificial pancreas
PID	Proportional-integral-derivative
MPC	Model predictive controllers
RL	Reinforcement learning
GEZI	Glucose effectiveness at zero insulin concentration
PPO	Proximal Policy Optimization
BG	Blood glucose levels
TIR	Time In Range
CVGA	Continuous Variability Grid Analysis
BraTS	Brain Tumor Segmentation
GANs	Generative Adversarial Networks
AdaBoost	AdaBoost Classifier
BGL	Blood Glucose Level
BGV	Blood Glucose Variability
CAN	Cardiac Autonomic Neuropathy
DTC	Decision Tree Classifier
F1	F1 score
Gaussian	Gaussian Naive Bayes
HR	Heart Rate
HRV	Heart Rate Variability
KNN	K-Nearest Neighbors Classifier
LR	Logistic Regression
MLP	Multi-Layer Perceptron Networks
RF	Random Forest
ROC	Receiver Operating Characteristic
SVM	Support Vector Machines
RL	Reinforcement Learning

1. Introduction

The convergence of artificial intelligence and healthcare has opened unprecedented opportunities for building intelligent systems that support clinical decision-making, optimize treatment strategies, and personalize patient care. In this dissertation, I present novel machine learning methodologies for modeling and automating medical processes, with a particular focus on diabetes management and medical image analysis. These two domains—though distinct—share a common thread: the need for data-driven, adaptive, and reliable systems capable of operating in real-world conditions.

The growing availability of medical data from wearable devices, imaging modalities, and continuous monitoring systems has enabled the development of advanced computational models that can learn from patient-specific patterns. Motivated by this potential, my research investigates both classical and deep learning methods to support and automate processes critical to diagnosis and therapy. My overarching goal is not only to advance the state of the art in algorithmic performance, but also to translate machine learning insights into clinically relevant tools.

This dissertation is structured around two primary application areas:

1. Diabetes Informatics

In this domain, I address three interconnected challenges:

- *Physical activity detection*, using heart rate and continuous glucose monitoring (CGM) data, to inform personalized insulin dosing and lifestyle recommendations.
- *Food intake detection*, by recognizing eating gestures from wearable motion sensor data, which contributes to accurate carbohydrate logging and improved dietary adherence.
- *Insulin therapy optimization*, via reinforcement learning models that learn adaptive dosing policies using CGM data alone, contributing to the development of closed-loop artificial pancreas systems.

2. Medical Image Analysis

In this domain, I focus on:

- *Infant brain tissue segmentation* from MRI scans using 2D, 3D, and spatiotemporal U-Net variants, which supports neurodevelopmental assessment during early stages of life.
- *Brain tumor classification*, by developing an L-net architecture that unifies segmentation and classification tasks to distinguish glioma, meningioma, and pituitary tumors with high accuracy.

Across these contributions, my work emphasizes methodological rigor, practical deployment considerations, and validation on real-world datasets. The models are evaluated using widely accepted metrics and benchmarked against existing methods to highlight their strengths and limitations.

By integrating clinical needs with machine learning innovations, this dissertation contributes to the broader goal of transforming healthcare into a more intelligent, proactive, and personalized discipline. I hope this work serves as a foundation for future research at the intersection of AI and medicine, and as a practical step toward data-driven decision support in both therapeutic and diagnostic workflows.

1.1 Diabetes

Diabetes mellitus is a chronic metabolic disorder characterized by elevated blood glucose levels resulting from defects in insulin secretion, insulin action, or both [R1]. Among the various types, **Type 1 Diabetes Mellitus (T1DM)** is an autoimmune condition that typically manifests in childhood or adolescence, where the immune system mistakenly destroys the insulin-producing beta cells in the pancreas [R2, R3]. This leads to absolute insulin deficiency, requiring lifelong insulin therapy. The condition can develop rapidly and is often detected due to sudden symptoms such as frequent urination, extreme thirst, unintended weight loss, and fatigue. Without timely intervention, T1DM can result in life threatening complications such as diabetic ketoacidosis.

In contrast to Type 2 diabetes, which develops due to insulin resistance and is often linked to lifestyle factors, T1DM is primarily genetic and immunological in origin. However, the global rise in diabetes cases across all types is undeniably influenced by modern lifestyles characterized by poor diet, sedentary behavior, and increased stress. These factors contribute significantly to the progression from prediabetes to full blown diabetes in genetically predisposed individuals.

Recent data suggest a concerning increase in the global prevalence of diabetes. Over the past four decades, more than 320 million people worldwide have been diagnosed with the disease [R4, R5]. In Hungary alone, a 2022 survey revealed that over 1.1 million individuals are affected by diabetes or prediabetes [R6, R7], with projections indicating further growth. The primary contributing factors include excessive consumption of sugars and processed foods, physical inactivity, and genetic susceptibility [R4, R5, R6].

Diabetes disrupts the body's ability to properly utilize glucose, the primary energy source for cells. Under normal physiological conditions, insulin—a hormone produced by beta cells in the islets of Langerhans in the pancreas—facilitates glucose uptake into cells by activating specific glucose transporters (notably GLUT-4) located in the cell membrane [R8, R1]. Once inside, glucose is either used immediately for energy production via the St. George-Crebs cycle or stored as glycogen for later use.

These insulin-responsive GLUT gates are predominantly present in the liver, muscle, and adipose tissue, the major sites of glucose metabolism. Insulin thus plays a pivotal role not only in glucose regulation but also in broader metabolic pathways. When insulin binds to its receptor on the cell surface, it initiates a cascade of reactions that result in glucose uptake, thereby lowering blood sugar levels and supplying the cell with energy [R8].

Glucose homeostasis in the human body is tightly regulated by hormones. While insulin lowers blood glucose levels, its antagonist, glucagon, increases them by promoting gluconeogenesis in the liver—i.e., the synthesis of glucose from non-carbohydrate sources [R9, R3].

Several types of diabetes exist, classified based on etiology and treatment approach:

- Type 1 diabetes mellitus (T1DM)
- Type 2 diabetes mellitus (T2DM)
- Maturity-Onset Diabetes of the Young (MODY)
- Gestational Diabetes Mellitus (GDM)
- Pancreatogenic diabetes
- Drug-induced diabetes

As T1DM is both prevalent and clinically distinctive, it was chosen as the focus of my research. This form of diabetes results in a complete lack of insulin due to immune-mediated destruction of pancreatic beta cells [R2, R3]. Exogenous insulin therapy is therefore essential for survival.

Normal fasting blood glucose levels in a healthy individual typically range between 4.0–5.4 mmol/L, rising to no more than 7.8 mmol/L two hours post-meal. In diabetics, these values can escalate, causing **hyperglycaemia**. Extremely high levels, such as above 25 mmol/L, can rapidly induce cellular dehydration, cognitive impairment, and potentially coma. Conversely, inappropriate insulin dosing may lead to **hypoglycaemia**, where glucose drops below 2.2 mmol/L, necessitating immediate intervention through rapid glucose intake to prevent seizures or unconsciousness [R10].

Hence, precise monitoring of blood glucose is essential. Insulin can be administered through syringes, pens, or insulin pumps. Regardless of method, effective insulin therapy hinges on regular glucose checks, which inform dosage adjustments and prevent extreme fluctuations.

Both T1DM and T2DM patients are typically advised to measure their blood glucose levels multiple times daily. This practice helps determine appropriate basal insulin dosing and is vital to avoid both hyper- and hypoglycaemia. In insulin pump users, continuous monitoring is preferred to align with the device's constant insulin delivery. This is best achieved using **Continuous Glucose Monitoring (CGM)** systems, which provide real-time glucose readings via a sensor inserted under the skin [R10].

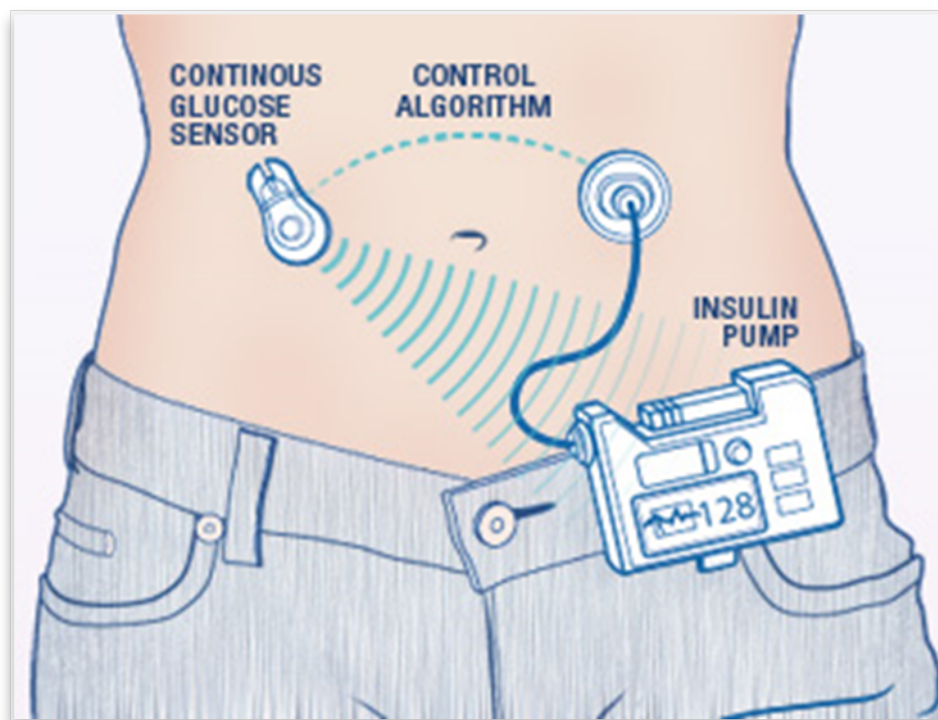


Figure 1.1: Artificial Pancreas [R11]

Unlike traditional fingerstick methods where glucose is measured via blood droplets using colorimetry or amperometry CGMs offer convenience and accuracy. They are especially beneficial during the early stages of therapy or when erratic blood glucose patterns are observed.

1.2 Magnetic resonance imaging

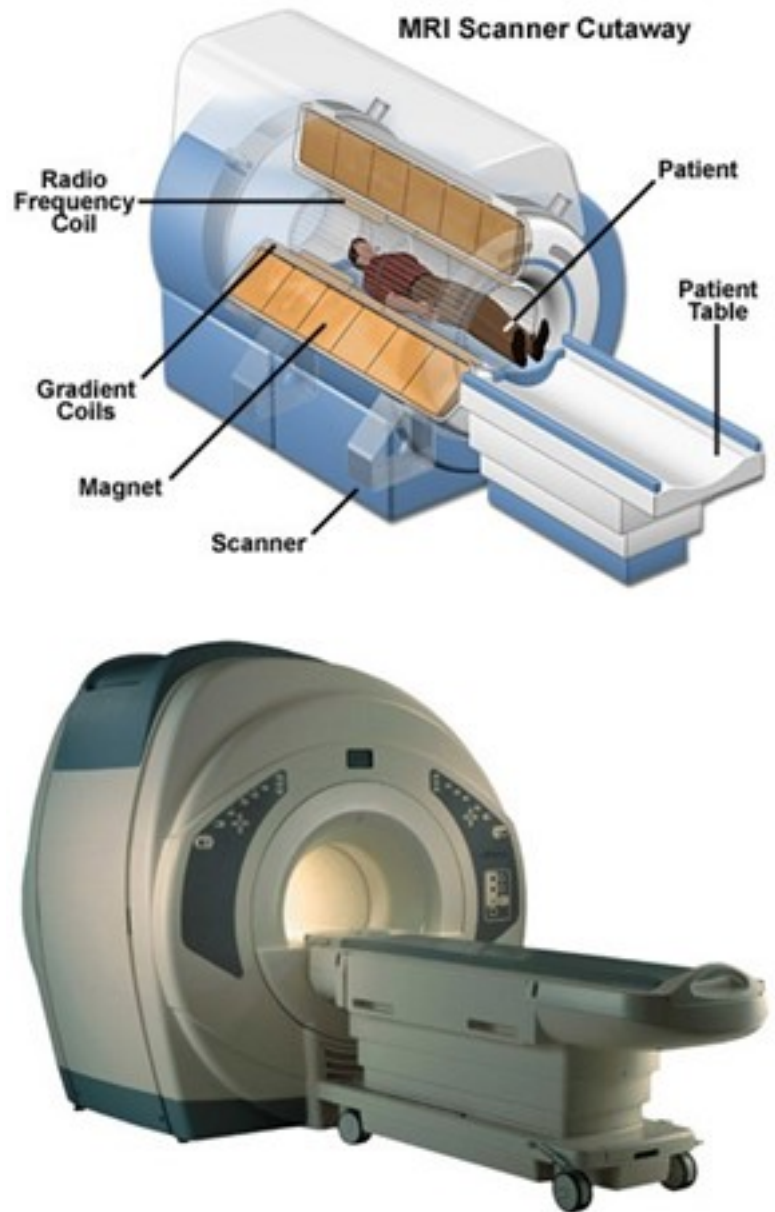


Figure 1.2: MRI machine [R12]

Magnetic Resonance Imaging (MRI) Figure 1.2 is one of the dynamically evolving modern imaging techniques that play a prominent role in medical diagnostics and scientific research. The first MRI image was presented to the scientific world in 1973 [R13], and four years later, in 1977, the first MRI image of a human body was published. Since then, MRI technology has evolved considerably and is now widely used in health care, disease diagnosis and research. The MRI technique is based on exploiting the magnetic properties of the hydrogen atoms in the body to produce images for detailed examination of the inside of the body. MRI allows doctors and researchers to obtain detailed images of organs, tissues and nervous structures without the need for surgical intervention. It thus contributes to the early detection and effective treatment of diseases and to the advancement of medical research. MR imaging is used to study changes in the magnetism of nuclei in a material, such as biological tissue [R14, R15]. The technique is based on the phenomenon of nuclear magnetic resonance (NMR), which is based on the idea

that when a nucleus with magnetic properties interacts with a strong magnetic field, the energy levels of the nuclei spin are split. The protons in living organisms have their own magnetic moment, as if they were small magnets. Because of this, they create a magnetic magnetic core resonance phenomenon that can be easily measured in the magnetic field. In the MRI process, these nuclei are manipulated by magnetic pulses and magnetic fields directed at them, and the magnetic signals emitted are detected and processed into images. The result is detailed and contrasty images of the area under investigation, which help doctors diagnose diseases and examine tissues in detail. MRI technology is an excellent way of imaging without the use of ionising radiation, making it a safe and very useful tool in modern medicine and research. Perhaps the best known MR imaging technique is proton MRI, which is based on the detection of the spin of hydrogen nuclei in water, fats, proteins and carbohydrates in humans and animals, mainly in two orientations (parallel or antiparallel). Placed in a stationary magnetic field (B_0 , the direction of the magnetic field of the MR machine), the spins are aligned parallel (low energy) or (a smaller fraction, one millionth of a millionth) antiparallel (high energy) to the direction of the magnetic field. The spin of the nuclei precesses (rotates around its own axis) with Larmor frequency around the magnetic field B_0 induced. The appearance of the reconstructed structural MR image, such as brightness and contrast, is influenced by a number of factors. These include the water-to-fat ratio of the region under investigation, which is a tissue characteristic and depends on the subject's health status. In addition, the density and movement of protons, the relaxation time of the tissue, as well as the sequence settings and contrast agents used, if any, also play a significant role in the formation of the images [R16]. The MR image Figure 1.3 is called T1- or T2-weighted, depending on whether the contrast of the image is mainly determined by the T1 or T2 relaxation time. In a T1-weighted image, the white matter will be of high signal intensity, the grey matter of medium signal intensity and the CSF of low signal intensity, whereas in a T2-weighted image, the white matter will be of low signal intensity, the grey matter of medium signal intensity and the CSF of high signal intensity. The appearance of pathological lesions is mainly determined by their physicochemical properties [R14, R15]. These image parameters and contrasts allow doctors to differentiate between different tissue regions of the body and to map possible lesions during MRI scans. Such information is extremely useful for diagnosis and disease monitoring.

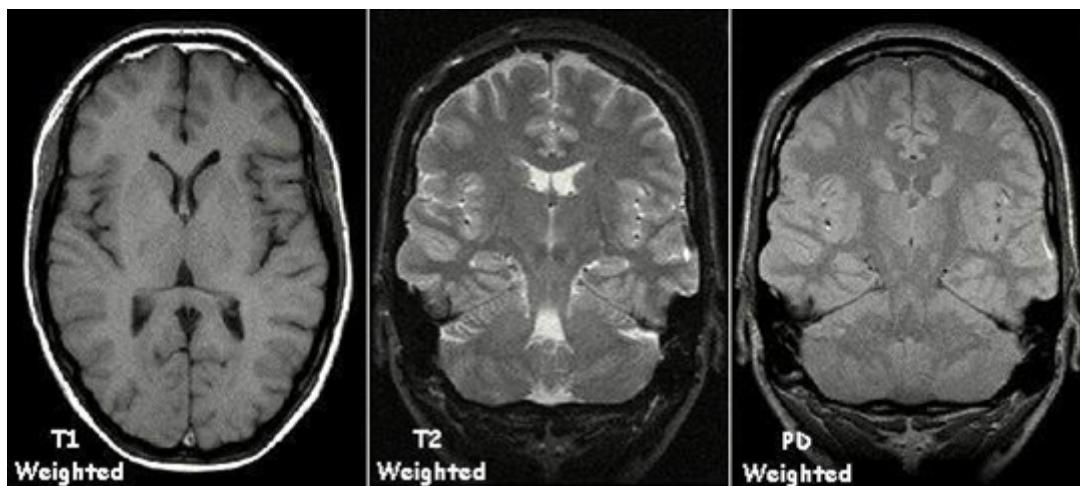


Figure 1.3: MRI captures types [R17]

These image parameters and contrasts allow doctors to differentiate between different tissue regions of the body and to map possible lesions during MRI scans. Such information is extremely useful for diagnosis and disease monitoring. One such type of MRI is functional MRI (fMRI), which allows the activity of the brain areas being studied to be tracked. fMRI helps to monitor brain functions such as speech, movement or memory continuously and to map brain

activity. MRI can also be used to obtain metabolic information, such as MR spectroscopy. This technique helps to study metabolic processes and identify the presence of different substances in tissues. Other specialised types, such as diffusion-weighted imaging (DWI) and diffusion tensor imaging (DTI), provide information on the diffusion movement of water molecules in tissues. These techniques can be useful in diagnosing brain injuries or neurological diseases, for example. MRI is therefore a versatile tool for studying not only the anatomy of the body, but also its functions, metabolism and diffusion processes, which contribute to more accurate and complex medical diagnoses. To summarise, the measurement of MR signal intensity, the accurate spatial localisation of signals of different intensities from different parts of the body, and the representation of signal intensity using a grey scale are the basis of MR imaging. Thanks to the increasing use of imaging techniques, complete MRI atlases are now available not only for humans but also for many animal species, such as the dog [R18].

1.2.1 Advantages of MRI

Magnetic resonance imaging does not use ionising radiation, so its harmful effects on the body are currently unknown. Therefore, as far as possible, MRI is recommended in all cases where unnecessary radiation exposure can be avoided. MRI offers many advantages:

- It has good spatial and temporal resolution.
- It is an objective test method suitable for both static and dynamic tests with standard techniques and documentation.
- Intravenous contrast administration is less frequently required (25%).
- It can be used to examine any organ, allowing the investigation of many types of diseases and reactions to certain drugs, as well as the early, non-invasive diagnosis of certain diseases.
- Because of the safety of the tests, a sufficient number of subjects can be collected and the tests can be extended to include juvenile subjects.
- Even if conventional and/or less expensive imaging techniques cannot establish a diagnosis, it is still worthwhile to perform an MR scan.
- The combination of different MR modalities has good differential diagnostic ability.
- Functional MRI is capable of replicating the results obtained with positron emission tomography (PET) and is currently more accessible than PET for (veterinary) medical practice in our country. The use of functional MRI instead of PET replaces the very expensive use of a cyclotron for isotope production. fMRI can be used to localise brain activity in near real time, which is useful before or even during surgery.

1.2.2 Disadvantages of MRI

- MRI is more expensive than the commonly used procedures, costing 50 000-60 000 HUF per scan (which only covers the costs), so it is still a diagnostic method with a low level of use in practice.
- There are currently relatively few MRI machines in our country, mainly for human examinations, which further complicates its use for veterinary purposes.
- Subjects undergoing MR examination must comply with safety requirements due to the high magnetic field (not applicable in case of implants or accidental metallic materials).

- The spatial resolution of fMRI does not allow measurements at the cellular or cell group level.
- Detection during fMRI is slow compared to neural activity.
- Different brain areas may have different hemodynamic responsiveness, and detailed regulatory processes are still intensively studied. fMRI measures brain activity only indirectly (indirect signal), since it can only examine the circulatory response to neuronal activity.

2. Performance Evaluation Metrics for Machine Learning and Deep Learning Models

In this dissertation, a standardized set of evaluation metrics is applied consistently across all experimental studies and analytical comparisons [R19]. Introducing these metrics comprehensively at the outset enables a unified and coherent interpretation of results and avoids unnecessary repetition in subsequent sections.

Within the fields of machine learning and artificial intelligence, evaluation metrics are indispensable for assessing model performance. They provide essential insights into how well a predictive system generalizes to unseen data, handles imbalanced classes, and aligns with real-world expectations. These metrics are not merely numerical summaries—they serve as critical tools for validating the correctness, reliability, and robustness of models. Consequently, this chapter presents each metric employed in this dissertation in detail, including their theoretical foundation, computational formulas, practical relevance, and illustrative visualizations.

2.1 Area Under the Curve (AUC) and ROC Curve

To evaluate classifier performance, two key metrics are used: **accuracy** and the **Area Under the Curve (AUC)** [R20]. In addition to these, Receiver Operating Characteristic (ROC) curves are employed for a more comprehensive analysis [R21]. While accuracy and AUC reduce classifier performance to a single scalar value, the ROC curve provides a richer representation by illustrating the trade-off between the true positive rate (sensitivity) and the false positive rate at various decision thresholds.

Each point on the ROC curve corresponds to a different classification threshold, making it a powerful tool for examining how a model performs under varying operational settings. The AUC metric, which calculates the area under this curve, reflects the model’s ability to distinguish between classes—an AUC of 1.0 denotes a perfect classifier, whereas 0.5 represents random guessing.

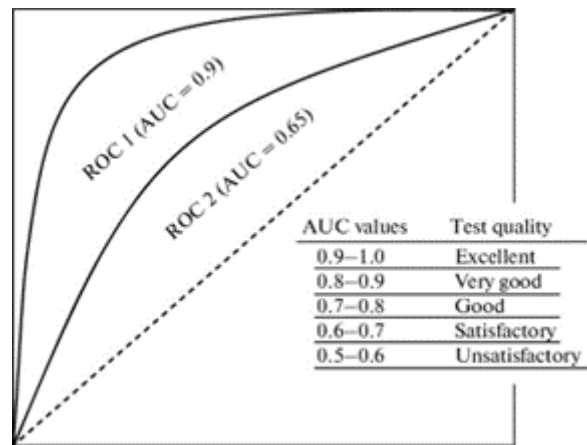


Figure 2.1: AUC metric [R22]

2.2 Confusion Matrix

The **Confusion Matrix** [R23] is a foundational tool in classification evaluation. It presents a tabular summary of the prediction results of a classification algorithm, with actual labels on one axis and predicted labels on the other. Each cell indicates the frequency of a particular combination of actual and predicted outcomes.

This matrix enables the calculation of several performance indicators, including recall, precision, specificity, and F1-score. For each test sample, the true label and predicted label are compared, and the outcome is recorded accordingly. An ideal classifier would produce a confusion matrix with values concentrated along the main diagonal, indicating perfect agreement between predictions and ground truth.

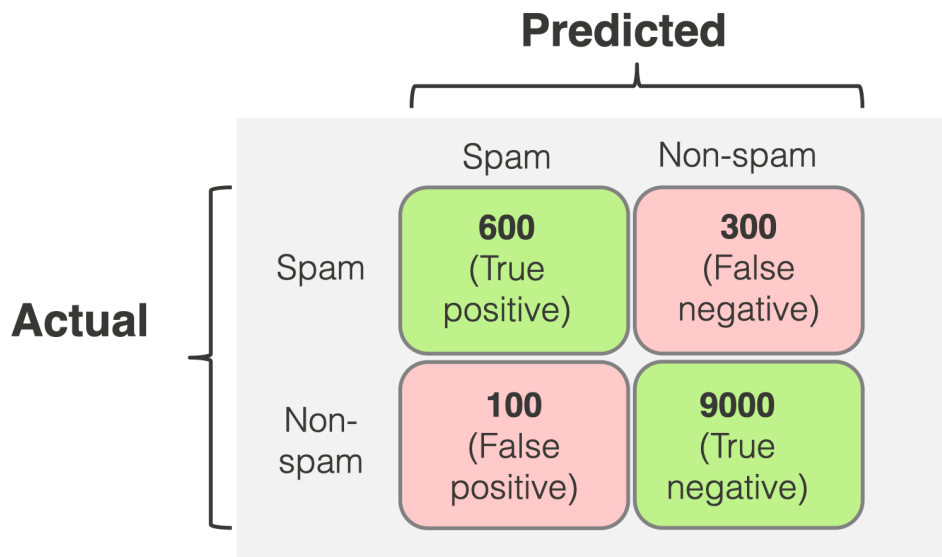


Figure 2.2: Confusion Matrix [R24]

As shown in Figure 2.2, a binary classifier yields a matrix with four fundamental components:

- True Positives (TP)

- False Positives (FP)
- False Negatives (FN)
- True Negatives (TN)

2.3 Derived Metrics: Recall, Precision, Accuracy, FPR, F1 Score, Specificity

To derive more nuanced insights from the raw output of classification models, several secondary metrics are computed based on the values in the confusion matrix. These metrics are especially important in real-world applications where the class distribution is often imbalanced, and different types of errors carry different consequences. The following subsections detail each of these evaluation metrics:

- *Recall (Sensitivity, True Positive Rate)*

Recall measures the model's ability to correctly identify all relevant instances of the positive class. It is defined as the ratio of true positives to the sum of true positives and false negatives:

$$Recall = \frac{TP}{TP + FN}$$

High recall is essential in scenarios where missing a positive case is costly or dangerous—for instance, in medical diagnoses (e.g., detecting cancer or diabetic complications) where false negatives can lead to missed treatment opportunities.

- *Precision (Positive Predictive Value)*

Precision quantifies how many of the predicted positive instances are actually positive. It is given by:

$$Precision = \frac{TP}{TP + FP}$$

This metric is particularly valuable when the cost of a false positive is high. For example, in spam detection, high precision ensures that legitimate emails are not misclassified as spam.

- *Accuracy*

Accuracy reflects the overall correctness of the model across both classes. It is calculated as the ratio of correctly predicted instances (both positive and negative) to the total number of predictions:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Although widely used, accuracy can be misleading in imbalanced datasets where one class dominates. In such cases, a high accuracy might still correspond to poor predictive power for the minority class.

- *False Positive Rate (FPR)*

The false positive rate indicates the proportion of actual negative cases that were incorrectly classified as positive:

$$FPR = \frac{FP}{FP + TN}$$

This metric is critical in domains where false alarms have significant consequences, such as fraud detection, where incorrectly labeling a legitimate transaction as fraudulent may lead to customer dissatisfaction.

- *F1 Score*

The F1 Score is the harmonic mean of precision and recall:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

It is especially useful when a balance between precision and recall is needed, and when the class distribution is uneven. A high F1 score indicates that the classifier performs well in identifying both positive samples and minimizing false alarms.

- *Specificity (True Negative Rate)*

Specificity measures the proportion of actual negative cases that were correctly classified:

$$Specificity = \frac{TN}{TN + FP}$$

This metric complements recall and is important in contexts such as disease screening, where it helps assess the model's ability to correctly rule out healthy individuals and avoid unnecessary treatments or tests.

2.4 Boxplot

The **Boxplot** [R25] is a graphical representation of the distribution of a dataset. It visualizes key descriptive statistics: the median, quartiles, and potential outliers. The position and size of the box help assess skewness, spread, and symmetry of the data, providing an intuitive overview of performance variation across different model runs or experimental conditions.

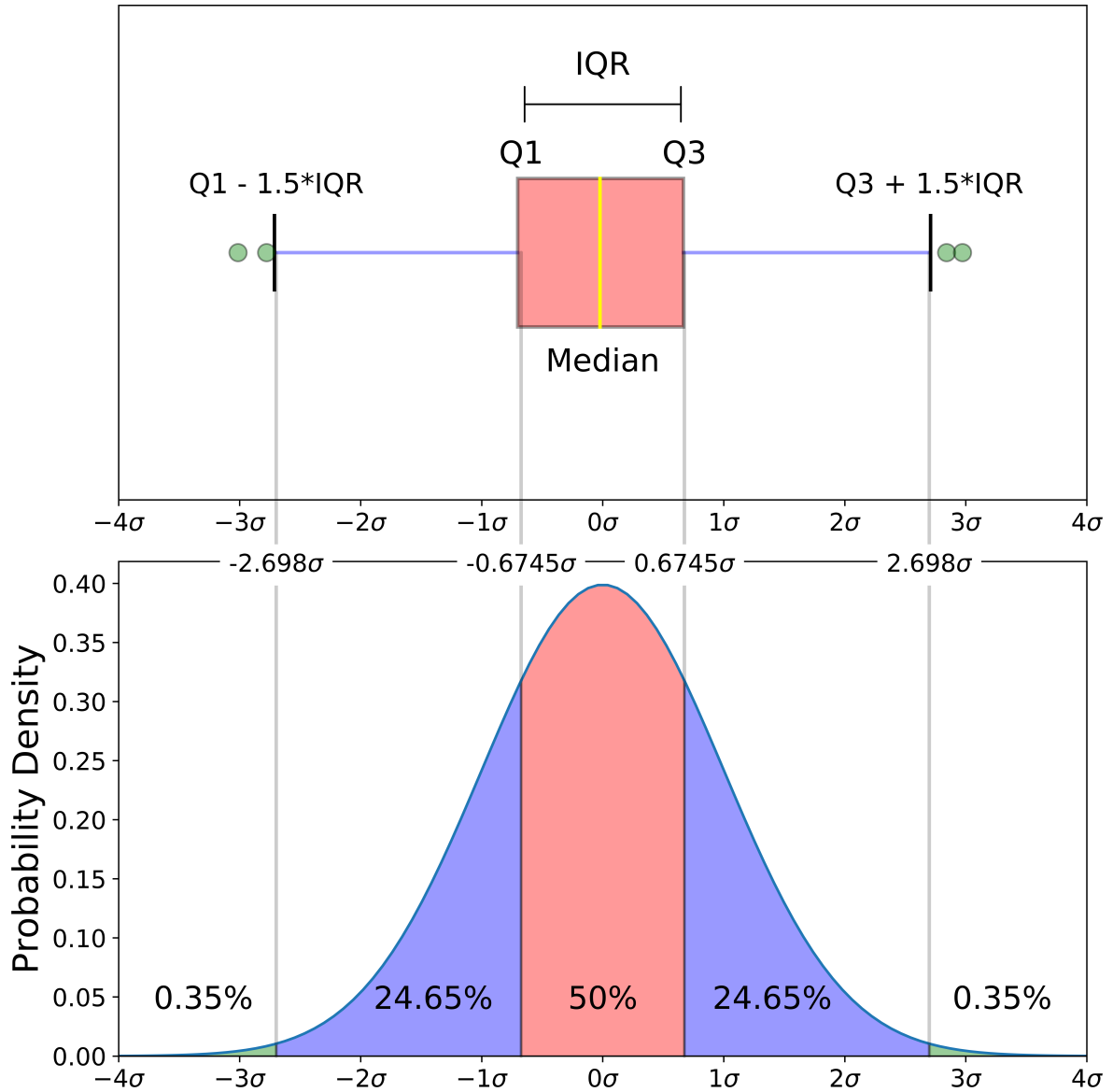
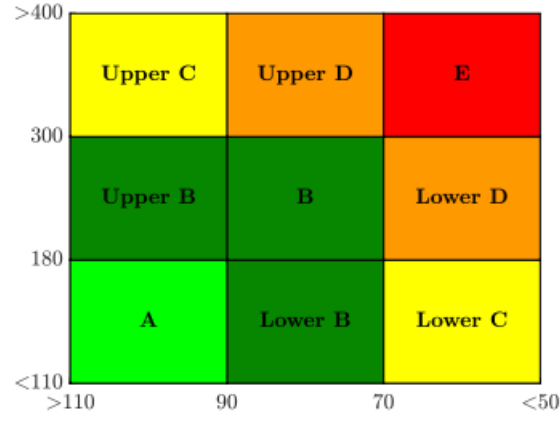


Figure 2.3: Boxplot [R26]

2.5 Control Variability Grid Analysis (CVGA)

The **Control Variability Grid Analysis (CVGA)** [R27] is used to evaluate extreme variations in blood glucose concentrations. The X-axis represents the minimum, and the Y-axis the maximum blood glucose value within the monitored interval. Each point on the graph typically corresponds to a subject or a specific monitoring period.

The plot is divided into nine zones that represent varying degrees of risk. Zone A is considered the safest and is centered around a minimum glucose level of 90 [mg/dl], ensuring optimal regulation. This tool is particularly useful for analyzing control strategies in diabetes treatment.



A Optimal control	$X=110-90$ [mg/dl] $Y=110-180$ [mg/dl]	
Lower B Slight deviation towards hypoglycaemia	$X=90-70$ [mg/dl]	$Y=110-180$ [mg/dl]
B Slight discrepancy	$X=90-70$ [mg/dl]	$Y=180-300$ [mg/dl]
Upper B Slight deviation towards hyperglycaemia	$X=110-90$ [mg/dl]	$Y=180-300$ [mg/dl]
Lower C Hyperglycaemia overcorrection	$X < 70$ [mg/dl]	$Y=110-180$ [mg/dl]
Upper C Overcorrection of hypoglycaemia	$X=110-90$ [mg/dl]	$Y > 300$ [mg/dl]
Lower D Inadequate treatment of hypoglycaemia	$X < 70$ [mg/dl]	$Y=180-300$ [mg/dl]
Upper D Inadequate treatment of hyperglycaemia	$X=90-70$ [mg/dl]	$Y > 300$ [mg/dl]
E Incorrect regulation	$X < 70$ [mg/dl]	$Y > 300$ [mg/dl]

Figure 2.4: CVGA diagram and zone description

2.6 Time in Range (TIR)

Time in Range (TIR) [R28] refers to the percentage of time that a patient's glucose levels remain within a target range—typically 70 to 180 [mg/dl]. This metric offers a comprehensive view of glycemic control and is widely used in clinical diabetes management.

Time Below Range (TBR) and *Time Above Range (TAR)* capture instances of hypoglycemia and hyperglycemia, respectively. For optimal regulation, it is generally recommended that TIR exceed 70% and TBR remain below 4%.

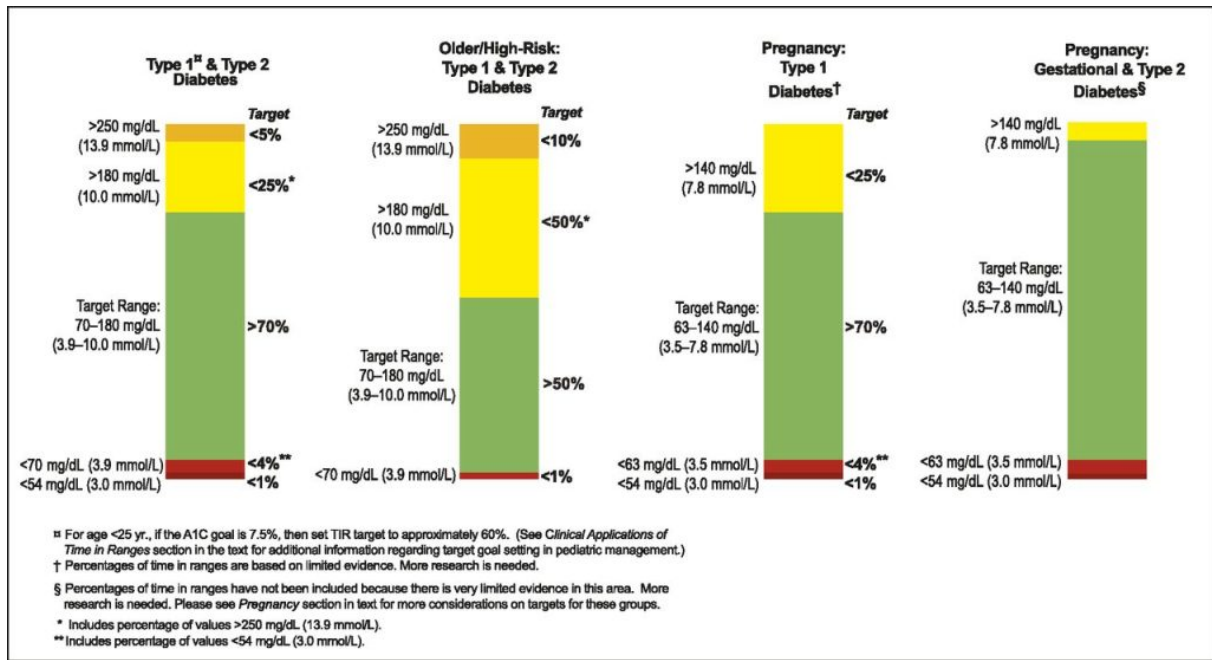


Figure 2.5: Time in range plots [R29]

3. Diabetes-related research

Thesis group 1: Diabetes research

Thesis I.1

I developed methods to detect physical activity of diabetes patients using artificial intelligence methods.

Thesis I.1.1

I have developed different methods using conservative artificial intelligence methodologies to detect physical activity in case of synthetic patient data. The results shown that recognition of physical activity in this scenario is possible using only blood glucose level.

Thesis I.1.2

I have developed methods to recognize physical activity in case of real patient data using artificial intelligence solutions. I have compared the results to the results of Thesis I.1.1 and it turned out that in case of real data the usage of only the blood glucose level is not sufficient. I successfully extended the physical activity detection solutions to consider heart rate data as well. Combining blood glucose and heart rate data provided satisfactory rate of recognition of physical activity.

Thesis I.1.3

I have demonstrated that using recurrent neural networks can produce much better results than simple machine learning algorithms to detect physical activity while they consider the blood glucose level data, heart rate data, and other derived modalities. I have proved that these networks do not require any preprocessing.

Thesis I.2

I have developed an artificial intelligence-based solution that can detect eating gestures using only accelerometer data. This solution is useful to determine when the patient is taking in carbohydrates, which knowledge is essential for any dietary decision support system or blood glucose control systems.

Thesis I.2.1

I have demonstrated that it is possible to create a personalised gesture detection model for the patients based on the given patient's dataset. According to the tests, this approach is sufficient to provide sophisticated results in eating gesture recognition for the given patient.

Thesis I.2.2

I have demonstrated that it is sufficient to use only one day of data to create an artificial intelligence-based method that can detect eating gestures with accurate results.

Thesis I.3

I have developed reinforcement learning-based blood glucose control methods for diabetes patients. The developed solutions use only blood glucose values from continuous blood glucose sensors with satisfactory results according to the defined metrics.

Thesis I.3.1

In experiments using several models, I have demonstrated that the PPO model performs best for the blood glucose regulation problem and administering the best working insulin schemes to regulate blood glucose level. I have demonstrated even if the model was trained only on a limited time horizon in terms of data, it successfully extrapolates the control to a larger time horizon if the training data covered a good representation of blood glucose data under regular conditions.

Thesis I.3.2

I demonstrated that the usage of the mixture of bump reward function and ELU activation function in the last layer of neural networks provides the best results for blood glucose regulation. I also proved that deeper networks performed better in this case without overfitting.

Publications relevant to the theses: [C1, C2, C3, C4, J1, C5, J2, C6, J3, J4].

Diabetes mellitus is a chronic disease affecting millions worldwide, characterized by dysregulation of blood glucose levels due to insulin deficiency or resistance. Managing diabetes requires continuous monitoring, timely insulin administration, and tracking of lifestyle factors such as physical activity and dietary intake. In recent years, intelligent systems powered by machine learning have shown promise in assisting both patients and healthcare providers in managing these complex variables more effectively.

This chapter presents my research on developing machine learning solutions for various diabetes related challenges. The focus is on creating intelligent, patient-centered models that support personalized monitoring and automated decision making. The systems developed here aim to enhance glucose control, reduce patient burden, and contribute to the broader goal of building an artificial pancreas system.

3.1 Chapter Overview

The chapter is organized into three major research directions:

- **Physical activity detection:** Accurate recognition of physical activity is critical for understanding blood glucose fluctuations and for safe insulin dosing. I developed models that detect physical activity from synthetic and real world data using a combination of CGM (Continuous Glucose Monitoring) and heart rate signals. Both traditional ML models and advanced recurrent neural networks (RNNs) were employed.
- **Gesture detection for food intake:** Carbohydrate intake timing is a crucial input for insulin regulation. I designed an activity recognition pipeline using data from inertial sensors (accelerometers, gyroscopes) to automatically detect eating gestures. The models are optimized for deployment in wearable devices such as smartwatches, making real world use feasible.
- **Reinforcement learning for insulin control:** I proposed a reinforcement learning based control system to automate insulin dosing decisions using only CGM data. The system learns optimal insulin policies through simulated interactions with a virtual patient, offering a foundation for closed loop glucose regulation.

3.2 Novel Contributions

The innovative aspects of this research chapter include:

- **Multimodal data integration:** The physical activity detection models integrate CGM and heart rate data, highlighting how heterogeneous data sources can complement each other in improving performance, especially under real life conditions.
- **Recurrent neural networks for temporal modeling:** I demonstrate that RNNs, particularly GRU and LSTM architectures, outperform classical ML models for detecting activity patterns from time series data. This is especially relevant in noisy, real world environments.
- **Wearable based food intake recognition:** The proposed solution uses only accelerometer data, eliminating the need for invasive glucose or dietary logging. Personalized models were shown to perform well, supporting individual patient calibration without requiring a universal model.

- **Reinforcement learning framework:** The insulin control system uses policy and value networks trained via Proximal Policy Optimization (PPO) on virtual patients. The controller achieves high Time-in-Range (TIR) glucose regulation and adapts to different metabolic conditions.
- **Validation on real and synthetic datasets:** All models were evaluated on both synthetic simulations and real patient datasets (e.g., Ohio T1DM, D1namo). Extensive metrics such as AUC, F1-score, precision, and TIR were used to benchmark performance, revealing robust and clinically relevant results.

In summary, this chapter offers an end-to-end demonstration of how machine learning can assist in various stages of diabetes management from lifestyle tracking to autonomous insulin dosing. The solutions presented are not only technically novel but also tailored for practical deployment in wearable devices and clinical systems.

3.3 Physical activity detection

3.3.1 Goal

The main objective of my research was to investigate whether it is possible to use machine learning based solutions to detect changes in blood glucose levels during physical activity and whether additional data are needed for this purpose. I generated one of the datasets *insilico*, taking into account two important aspects. The first aspect was to be able to produce the data set in real life in patients with as few sensors as possible, while the second objective was to create a simulator to have a large amount of synthetic data available for future studies. Real measurements can be expensive, so building the simulator was of paramount importance. I used the simulated data set for the first experimental tests. I continued my research with two more real patient databases. I applied the experience gained here to the data generated by the simulator and also experimented with different machine learning algorithms. In addition, I analysed the results of the highly controlled patient measurements in the framework mentioned in the introduction of the project. I then extracted additional features using an additional detector and used these new feature vectors to perform further tests on real patient data. My first step was to integrate and extend the Jacobs simulator [R30] published on GitHub with the patient simulator. This simulator generates virtual type 1 diabetes patients and focuses on the external input of insulin dosing based on the given parameters, where the Cambridge model provided the basis [R31]. My primary goal was to extend this simulator with a continuous glucose monitoring sensor (CGMS) module that generates noisier, more realistic synthetic measurement data, and then use the integrated system to create a data set. I then perform a performance analysis using machine learning algorithms.

3.3.2 Introduction

Diabetes mellitus (DM) is a chronic metabolic disorder linked to the hormone insulin. Type 1 DM (T1DM) is an autoimmune disease that can arise abruptly and is influenced by genetic and other unidentified factors. In contrast, Type 2 DM (T2DM) typically develops gradually, with obesity and physical inactivity as significant risk factors. T2DM is often undiagnosed for a long time, with patients frequently discovering their condition through symptoms related to the disease [R32]. Physical activity is essential in diabetes treatment, benefiting both T1DM and T2DM patients. For individuals with T1DM, regular exercise enhances glycemic control [R33]. The intensity of exercise is also critical for these patients. High intensity interval exercise and training are considered safer than continuous exercise because they lower the risk of hypoglycemia [R33]. However, engaging in unplanned exercise can be hazardous if not managed properly during

insulin therapy. Specifically, insulin overdose may occur in individuals who fail to adjust their insulin doses for exercise or neglect to update insulin pump settings during pump therapy, leading to severe hypoglycemia episodes [R34]. For diabetics, hypoglycemia is a critical condition, as declining glucose levels can trigger ketoacidosis, potentially resulting in coma or even death in the short term. Therefore, it is crucial to carefully consider physical activity in daily life, especially with semi automated therapies like insulin pump applications [R35]. In automatic glucose control, control algorithms must consider the patient's physical activity. Subroutines that can detect exercise events are crucial for preventing hypoglycemic episodes, even if there are errors in reporting or calculations by patients. Exercise induced reductions in blood glucose (BG) levels occur with a slight delay, but the impact of physical activity on BG regulation can last up to 48 hours after exercise, depending on the intensity and duration of the activity, as discussed in [R36, R37]. Understanding the impact of various physical activities is essential for timely interventions in blood glucose control.

A significant challenge for researchers is developing algorithms that can detect unexpected physical activity to enhance decision making and treatment in partially automated blood glucose (BG) control systems. The difficulty in developing these algorithms arises from limited available data and the frequent lack of patient cooperation. However, there is a strong demand from patients and the industry for high-quality physical activity detection systems to support effective decision making, particularly in insulin pump therapy. Modern insulin pump systems are based on the artificial pancreas (AP) concept, which consists of three main components: a Continuous Glucose Monitoring System (CGMS) for tracking BG levels, an insulin pump for delivering insulin, and advanced control algorithms. Typically, AP systems integrate these elements [R38, R39]. When additional sensors are not available, such as body worn activity trackers or integrated accelerometers (Inertial Measurement Units, or IMUs, which include accelerometers and other motion sensors) or heart rate (HR) sensors within the CGMS or insulin pump, the only way to detect physical activity in users of these systems is through the CGMS signal. However, the main challenge is the delay in reflecting the effects of exercise in the CGMS signal. To address this limitation, IMU and HR signals can serve as valuable supplements to CGMS signals, as they can accurately indicate physical activity [R40, R41]. Recognizing, identifying, and categorizing physical activity by type and intensity are crucial components of effective T1DM management. Numerous solutions are available in this area, particularly those utilizing Inertial Measurement Unit (IMU) sensors, as discussed in [R42]. A recent advancement involves using IMUs specifically to detect and classify physical activity in diabetic patients, as highlighted in [R43]. The use of IMUs is particularly advantageous given the prevalence of cardiac autonomic neuropathy (CAN) in individuals with diabetes, characterized by autonomic nervous system (ANS) dysfunction and an elevated resting heart rate (HR) [R44].

Cardiac autonomic neuropathy (CAN), a common long-term complication in people with diabetes, can diminish the predictive accuracy of heart rate signals in patients with type 1 diabetes mellitus (T1DM) [R45]. However, the relationship between CAN, blood glucose (BG) levels, BG variability (BGV), and heart rate variability (HRV) over the short and medium term is not yet fully understood. Additionally, some studies indicate that the connections between CAN, HR, and HRV require further investigation [R46]. Most wearable activity monitors on the market do not allow users to access raw IMU data, as noted in [R47]. Although some devices, like the Empatica E4, do offer access to raw IMU data, their high cost (around 1000 USD) makes them less accessible for people with diabetes. Conversely, wearable sensors that provide heart rate (HR) data with a sampling interval of at least 5 minutes are more affordable and offer convenient data access, either directly from the device or through activity tracking apps, as highlighted in [R47]. This 5 minute sampling interval enables the use of HR data in combination with CGMS signals to assess physical activity. Artificial intelligence (AI) tools have shown their effectiveness in identifying patterns across various applications in biomedical engineering, as demonstrated by studies such as [R48, R49, R50, R51, R52]. These tools are

also useful in diabetes treatment, where AI has proven advantageous, as indicated in studies like [R53, R54, R55, R56]. Effectively managing T1DM requires a personalized approach to insulin therapy, diet, and physical activity. Monitoring physical activity is crucial for optimizing glycemic control. However, traditional methods often fall short in providing comprehensive insights, leading to the exploration of innovative solutions. In this context, Recurrent Neural Networks (RNNs) [R57] have emerged as a promising tool for detecting and analyzing physical activity patterns in individuals with T1DM [R58].

RNNs are specifically designed for processing sequential data and excel at recognizing temporal dependencies in human movement. This makes them ideal for identifying various physical activities, from simple tasks like walking to more complex exercises. The recurrent nature of RNNs allows them to capture dynamic changes in activity and distinguish nuances between different activities with remarkable precision, while also providing adaptability [R59].

In summary, the use of RNNs offers a powerful and efficient solution for accurately detecting and analyzing physical activity in T1DM patients. This approach has great potential for enhancing my understanding of individual activity patterns, leading to more personalized and effective T1DM management.

Based on the information and research presented earlier, it can be concluded that the CGMS signal, as well as the combination of CGMS and heart rate (HR) signals, are promising candidates for detecting physical activity. In this study, my goals include developing AI programs that can identify physical activity by analyzing either the CGMS signal alone or the combination of CGMS and HR data in a binary manner (determining the presence or absence of physical activity). These algorithms show significant potential, particularly in closed loop insulin delivery systems. It is important to note that, at this initial stage of my research, the focus is on recognizing the presence of physical activity without categorizing its type.

3.3.3 Synthetic Dataset

I employed synthetic data generated by an extended version of the Jacobs T1DM simulator [R60]. While the simulator utilizes the Cambridge model as its basis, it includes an embedded physical activity submodel, which is an extension compared to the version presented in [R61]. My study utilized the single hormone virtual patient population, where the simulator expects insulin as the sole control input. These virtual patients, totaling 20 in number, were identified and validated for Type 1 Diabetes Mellitus (T1DM) based on a 3.5 day outpatient Artificial Pancreas (AP) study [R60]. The simulator, originally published on GitHub (https://github.com/petejacobs/T1D_VPP), is open source. To enhance the realism of data generation, I integrated the Continuous Glucose Monitoring System (CGMS) model from [R62] into the simulator.

The simulator offers straightforward parameterization options for carbohydrate (CHO) and insulin intake. I maintained the default settings of the simulator and applied the following regimens, utilizing randomization for both the timing and quantities of CHO intake:

- The times of meal consumption randomly varied between -30 minutes and +90 minutes with respect to prescribed times. The default time instances in the simulator were: breakfast at 6 am, lunch at 12 pm and dinner at 6 pm.
- The amount of breakfast was set 35 ± 10 [g].
- The amount at lunch varied between 79 ± 10 [g]
- The amount at dinner varied between 117 ± 10 [g].
- The duration of physical activity varied between 30 minutes to 90 minutes.
- The blood glucose level at the beginning of the day varied between 160 ± 20 [mg/dL].

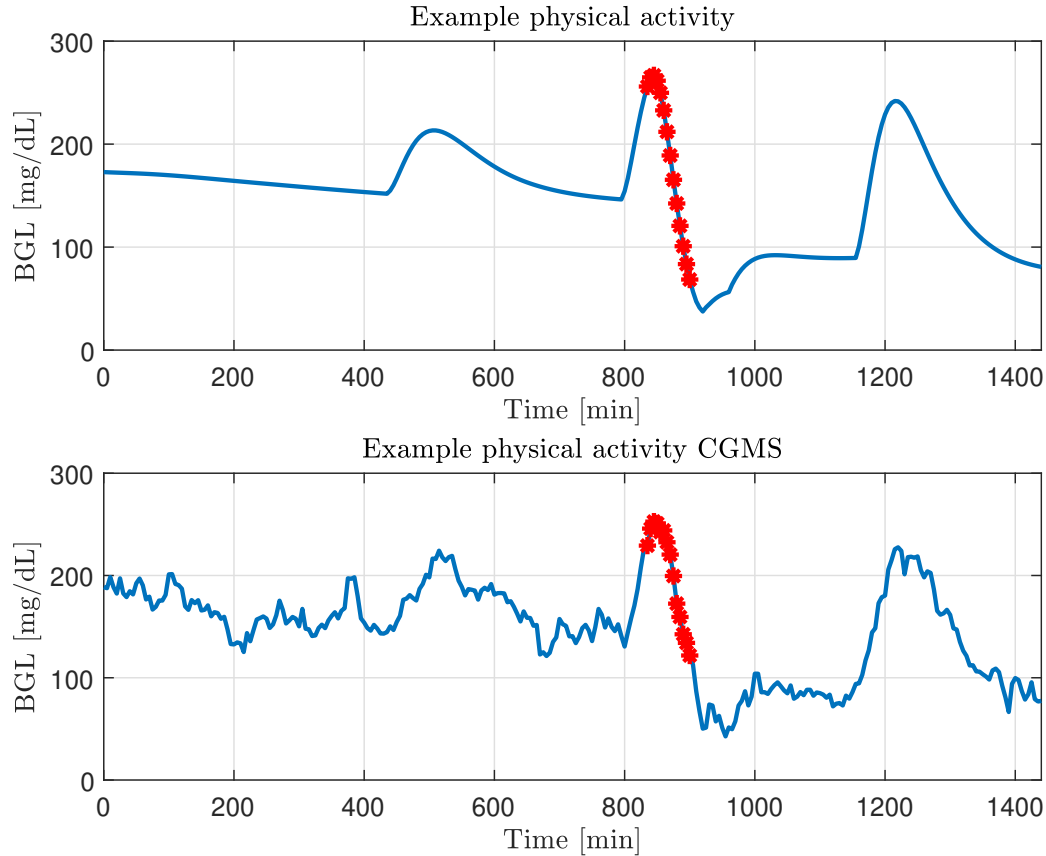


Figure 3.1: Jacobs simulator model output indicating the physical activity with the red section on figure, without CGMS (top) and with CGMS (bottom)

Incorporating the Continuous Glucose Monitoring System (CGMS) model alters the "pure" blood glucose (BG) output generated by the simulator. An example of this modification is illustrated in Fig. 3.1. I maintained a realistic sampling time, with the blood glucose level simulated by the CGM sensor measured every five minutes.

The synthetic data generation process proceeded as follows:

I randomly selected 13 virtual patients from the available pool of 20. Throughout the data generation process, I kept the parameters of the selected virtual patients unchanged.

For each patient, the simulation spanned 640 days. Rather than conducting a single extended simulation, each day was simulated separately over a 24-hour period, resulting in 288 sample points per day. I utilized a sliding window of 15 sample points to extract features. Samples within the sliding window were indexed from 0 to 14, yielding 274 data rows per day. Each sliding window was further divided into three smaller inclusive windows, each consisting of five consecutive samples with indexes 0 to 4, 5 to 9, and 10 to 14, respectively. From each sliding window, I extracted 32 features. The ground truth of the dataset is outlined as follows:

- The patient's body weight w ;
- End-to-end blood glucose level change d defined as

$$d = bg(14) - bg(0) \quad (3.1)$$

- The blood glucose level variation between consecutive sampled points, or in other words the first order differences of blood glucose levels, defined as

$$dp(i) = bg(i + 1) - bg(i) \quad (3.2)$$

for any $i = 0 \dots 13$;

- End-to-end blood glucose level change in all inclusive sliding windows, defined as

$$dpp(i) = bg(5i + 4) - bg(5i) \quad (3.3)$$

for any $i = 0 \dots 2$;

- Second order changes of the blood glucose level, computed from 3 consecutive samples with the formula

$$\begin{aligned} ap(i) &= dp(i + 1) - dp(i) \\ &= bg(i + 2) - 2 \times bg(i + 1) + bg(i) \end{aligned} \quad (3.4)$$

for any $i = 0 \dots 12$;

- The decision dc , which is used as ground truth throughout this study.

The entire dataset comprises 2,279,680 entries, with 95% representing measurements indicating no activity (0), and the remaining 5% indicating activity (1). Due to this significant imbalance, the problem can be classified as an anomaly detection task in machine learning.

3.3.4 Tested models on synthetic dataset

This study encompasses eight distinct machine learning algorithms, some of which exist in multiple versions owing to varying parameter settings, culminating in a total of 13 models. Below, I provide descriptions of these models along with their unique identifiers.

- *Logistic Regression* [R63], with maximum 1000 iterations, and L2-type penalty (LogReg);
- *AdaBoost Classifier* (Ada) [R64] with maximum 50 trees;
- *DecisionTree Classifier* (DecTree) [R65, R66] with unlimited tree depth and all decisions allowed to use any one of the features;
- *Gaussian Naive Bayes* (Gauss) [R67];
- *K-Nearest Neighbors Classifier* (KNN) [R68] using $k - d$ tree implementation [R69] and $k = 5$;
- *Support Vector Machines* [R70] with 1000 iterations in five kernel variants: radial basis function kernel (SVM1), sigmoid kernel (SVM2), 3rd degree polynomial kernel (SVM3), 5th degree polynomial kernel (SMV4) and 10th degree polynomial kernel (SVM5);
- *Random Forest* [R71, R72] with 100 trees (RF);
- *Multi Layer Perceptron Networks* [R73] with four hidden layers of sizes 100, 150, 100, and 50, respectively, maximum 1000 iterations, and three variants of activation functions: logistic (MLP1), ReLU (MLP2), and tanh (MLP3).

3.3.5 Results on synthetic dataset

Table 3.1 presents the confusion matrix for all tested classification models. Values presented in this table are normalized in each row. The classification can be called successful if the rate of both true positives and true negatives are high, typically above 0.8. Only the RF model achieved higher rate than 0.9 of both TP and TN, while KNN, AdaBoost, Decision Tree and all tested multi layer perceptron models scored at both normalized indicators above 0.8. The three SVM models that used polynomial kernel apparently predicted the opposite.

Table 3.1: Normalized confusion matrix obtained by the use of various classification methods

		Predicted value	
		0	1
True value	0	LogReg : 0.779	LogReg : 0.221
		Ada : 0.833	Ada : 0.167
		KNN : 0.980	KNN : 0.020
		DecTree : 0.991	DecTree : 0.009
		Gauss : 0.789	Gauss : 0.211
		SVM1 : 0.717	SVM1 : 0.283
		SVM2 : 0.743	SVM2 : 0.257
		SVM3 : 0.327	SVM3 : 0.673
		SVM4 : 0.311	SVM4 : 0.689
		SVM5 : 1.000	SVM5 : 0.000
		RF : 0.961	RF : 0.039
		MLP1 : 0.864	MLP1 : 0.136
		MLP2 : 0.863	MLP2 : 0.137
		MLP3 : 0.848	MLP3 : 0.152
	1	LogReg : 0.221	LogReg : 0.779
		Ada : 0.167	Ada : 0.833
		KNN : 0.101	KNN : 0.898
		DecTree : 0.166	DecTree : 0.834
		Gauss : 0.211	Gauss : 0.789
		SVM1 : 0.284	SVM1 : 0.716
		SVM2 : 0.257	SVM2 : 0.743
		SVM3 : 0.673	SVM3 : 0.327
		SVM4 : 0.689	SVM4 : 0.311
		SVM5 : 1.000	SVM5 : 0.000
		RF : 0.073	RF : 0.927
		MLP1 : 0.136	MLP1 : 0.864
		MLP2 : 0.137	MLP2 : 0.863
		MLP3 : 0.152	MLP3 : 0.848

Figure 3.2 illustrates the ROC curves of all classification models, while Figure 3.3 highlights the Area Under Curve (AUC) metric values, excluding those below 0.75. These figures collectively indicate that the Random Forest model exhibited excellent performance, boasting an AUC value of 0.98. However, it's worth noting that according to the classification benchmark by Trifonova et al. (2013) [R74].

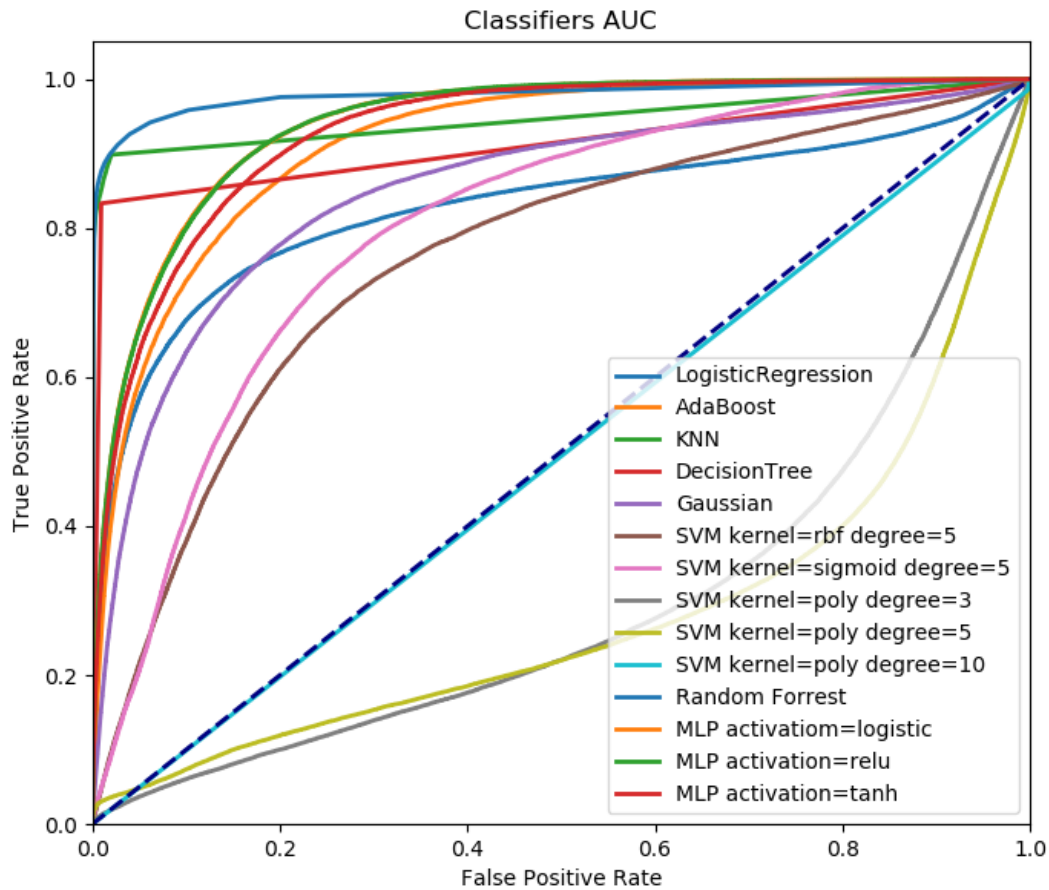


Figure 3.2: ROC curve of models

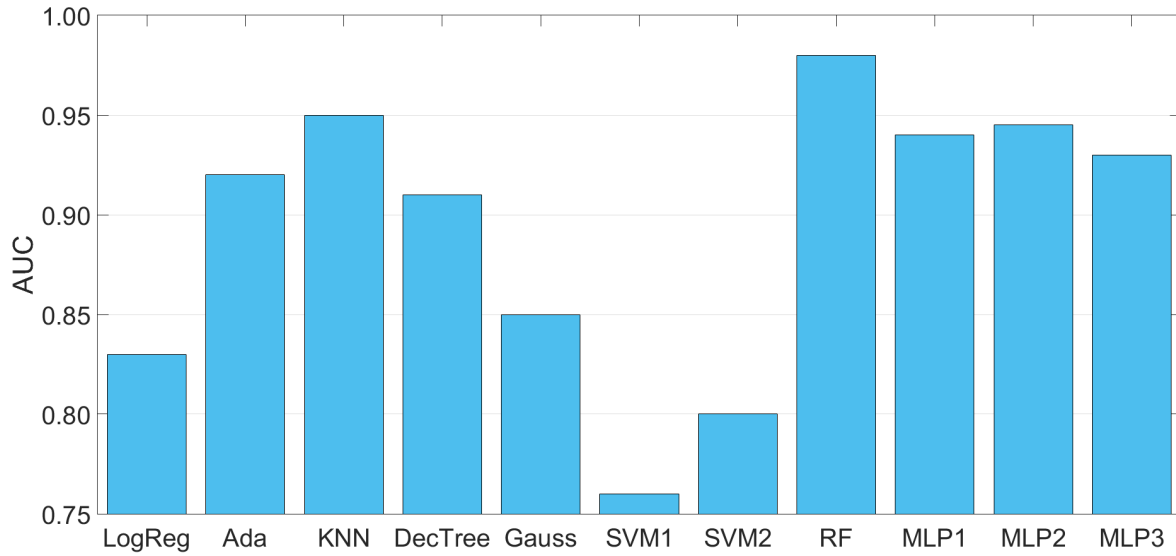


Figure 3.3: AUC values obtained by various classification models

3.3.6 Real Patient Datasets

Before presenting the data of real patients, it is important to note that in their case, blood glucose levels alone were no longer sufficient to achieve accurate results in contrast to the synthetic data, where this single feature yielded satisfactory predictive performance. Due to the complexity of real life scenarios, individual variability, and the influence of various physiological and lifestyle factors, the inclusion of additional features became essential to improve model accuracy. Among these, heart rate data proved particularly relevant, as it effectively reflects the body's current state and physiological responses. It is also important to emphasize that in the simulation environment that is, when working with synthetic data I did not have the opportunity to incorporate such complex physiological features, which inherently limited the realism of the simulations. Consequently, working with real data presented not only technical but also methodological challenges, which became one of the key takeaways of the research.

The Ohio T1DM Dataset

The Ohio T1DM dataset is a semi public dataset accessible to researchers investigating Type 1 Diabetes Mellitus (T1DM). Initially released in 2018 by Marling et al. [R75] for the Blood Glucose Level Prediction (BGLP) challenge [R76], it was later expanded in 2020 with data from an additional six subjects [R77, R78].

The dataset comprises 8 weeks of continuous data collected from 12 subjects diagnosed with Type 1 Diabetes Mellitus (T1DM). Throughout the monitoring period, each subject utilized a continuous glucose monitoring system (CGMS) and followed insulin pump therapy. The dataset for each patient includes a comprehensive range of information, such as CGM readings of blood glucose levels taken every 5 minutes, blood glucose levels obtained through periodic self monitoring of blood glucose (finger sticks), administered insulin doses (bolus and basal), self reported meal times along with carbohydrate estimates, self reported times of activities including exercise, sleep, work, stress, and illness, as well as physiological data from fitness bands and environmental data such as temperature [R78].

In my study, I extracted the following data types from the original dataset, which featured a complex XML structure:

- glucose level: contains the time stamps (date and time in: DD-MM-YYYY HH:MM:SS format) and the CGM data (in mg/dl) recorded every 5 minutes;

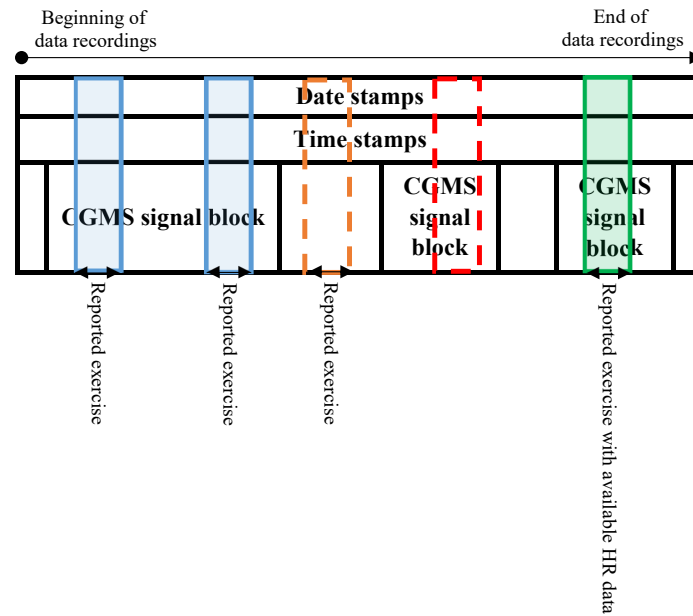


Figure 3.4: The ways of data extraction from the Ohio T1DM Dataset at a given patient – graphical example. The black arrows represent a 24 hours long block according to the time stamps, midnight to midnight. In this example, the blue regions belong to those 24 hours long blocks, where CGMS data were available and exercise was reported without available corresponding HR data. The green region represents a 24 hours long data block where CGMS data were available, exercise was reported and the corresponding HR data were available. The transparent area covered by orange dashed line represents self reported exercise (24 hours long block during which the exercise happened) without corresponding CGMS signal. The transparent area covered by red dashed line represents a 24 hours long block during which probably an exercise event happened but it was not reported (“non annotated outcome”).

- exercise: contains time stamps of the beginning of the exercise (date and time in: DD-MM-YYYY HH:MM:SS format), intensity values (subjective intensities on a scale of 1 to 10, with 10 the most physically active), type (exercise type), duration in minutes and a “competitive” field;
- basis_heart_rate: contains time stamps (DD-MM-YYYY HH:MM:SS) and the HR data (in bpm) recorded every 5 minutes;
- body weight of the patient expressed in kg (measured once at the beginning of the 8-week experiment).

The primary challenge associated with utilizing the Ohio T1DM dataset arises from the fact that all physical exercise is self reported. I conducted an in depth analysis of the extracted data fields, the results of which are illustrated in Fig. 3.4. Here, I present a graphical representation of my findings from the patient’s recordings. My analysis revealed distinct “islands” within the data structure: notably, the data and time stamps were consistently available throughout the duration of the experiment, while the CGMS data were organized into “blocks” with specified starting times and durations, often spanning multiple days. In contrast, exercise events were reported sporadically, with varying starting times and durations.

I identified exercise events where corresponding CGMS data were absent (e.g., the transparent area covered by orange dashed lines in Fig. 3.4). These data were neglected during extraction since my use cases necessitated the presence of the CGMS signal. Similarly, I excluded 24-hour intervals where no physical activity was reported, as my investigations revealed inaccuracies in patient self reporting, suggesting possible unreported physical activity (e.g., the transparent area covered by red dashed lines in Fig. 3.4). Consequently, the amount of data available for extraction significantly decreased. This reduction in data volume holds significance for later stages, as the algorithms developed in this study could be employed for data annotation.

To ensure that extracted data unequivocally reflected self reported exercise, I opted to extract only those smaller 24-hour regions (midnight to midnight) where both the CGMS signal and reported exercise were present (e.g., the blue and green regions in Fig. 3.4). These extracted data were utilized when the modeling objective was to detect physical activity solely from the CGMS signal. Additionally, I considered cases where physical activity was detected using the CGMS signal alongside corresponding exercise events where heart rate (HR) data with matching date and time stamps were available (e.g., the green region in Fig. 3.4). This type of data extraction further reduced the size of the usable dataset.

Thus, I have extracted two “cleaned” datasets:

1. CGMS data in 24 hours long blocks where self reported exercise happened within the 24 hours (e.g. Fig. 3.4 – blue and green regions). The exercise event was set to 1 where the exercise was ongoing. I made an exception – where the reported activity level was lower than 2, I set the exercise event to 0. The numerical properties of the extracted dataset can be seen in Table 3.2.
2. CGMS data in 24 hours long blocks where self reported exercise happened within the 24 hours and HR data were also available (e.g. Fig. 3.4 – green region). The exercise event was set to 1 where the exercise was ongoing. I made an exception – where the reported activity level was lower than 2, the exercise event was set to 0. The numerical properties of the extracted dataset can be seen in Table 3.3.

Each data record for each patient followed this structure: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), Blood glucose (BG) level from CGMS (concentration), Heart rate (HR) value (integer), Exercise indicator (0/1)]. These records were arranged chronologically. Weight data, reported only once at the beginning of the experiment, was managed separately.

Table 3.2: Properties of the cleaned data from the Ohio T1DM dataset used for patients' blood glucose feature extraction.

Ohio T1DM Patient (OP) ID	Number of records	Total time duration in days
OP3	546	2
OP8	2730	10
OP9	546	2
OP10	3276	12
OP14	2730	10
OP18	546	2
OP19	7917	28
OP20	819	3
OP21	1638	6
OP22	1365	5
OP23	4914	18
Total	27027	94

Table 3.3: Properties of the cleaned data from the Ohio T1DM dataset used for the extraction of patients' blood glucose and heart rate features.

Ohio T1DM Patient (OP) ID	Number of records	Total time duration in days
OP8	2730	10
OP9	546	2
OP10	3276	12
OP14	2730	10
OP18	546	2
OP19	7917	28
OP20	819	3
OP21	1638	6
Total	20202	71

The D1namo Dataset

The D1namo dataset consists of measurements from 20 healthy individuals and 9 subjects with Type 1 Diabetes Mellitus (T1DM), which were obtained using wearable devices (Zephyr Bio-Harness 3) under non-clinical conditions. This study exclusively focuses on the data from the 9 T1DM patients.

The dataset comprises electrocardiogram (ECG), breathing, and accelerometer data, alongside glucose measurements and annotated food pictures [R79]. Additionally, aggregated values derived from the ECG signals are included, which provide information on heart rate and activity level, both of which are utilized in my study. Furthermore, the patient's body weight is provided as part of the dataset.

The dataset generally spans a few days per subject (1-3 days with overlapping). While the sampling time of the Continuous Glucose Monitoring (CGM) is 5 minutes, the sampling frequency of heart rate (HR) and activity is 1 Hz. However, the data quality is not as robust as in other investigated datasets, exhibiting numerous inconsistencies and missing values in CGM, HR, and activity recordings.

The basis of my data cleaning involved considering blood glucose (BG) registrations in the format: [date stamp in YYYY-MM-DD, time stamp in HH:MM:SS, BG concentration in mmol/L, type of measurement (manual/CGMS)]. From the HR and activity data, I extracted values corresponding to the BG registrations based on date and time stamps. Consequently, for each subject, I obtained CGM data at a 5-minute sampling time, and HR and activity data at a 1-second sampling time.

The electrocardiogram (ECG) sensor operates at a sampling frequency of 250 Hz, and the accelerometer at 100 Hz, although the data stream is provided on a 1-second basis. As a result, approximately 300 data points of HR and activity level values were available between two BG registrations in the extracted data (with occasional data inaccuracies).

To address this issue, I aggregated HR and activity levels between BG registrations using the numerical average of the values ($\frac{1}{N} \sum_{i=1}^N \text{HR}_i$, where $N = 1.. \sim 300$, depending on data availability). The activity level, automatically determined by the Zephyr sensor from Inertial Measurement Unit (IMU) and HR data, is considered mild if it exceeds 0.2 according to US-ARIEM guidelines [R80]. Thus, I classified an exercise event corresponding to a given record as 1 if the activity level was higher than 0.2; otherwise, it was set to 0.

Following data cleaning, structured datasets were obtained. Similar to the Ohio T1DM dataset, the final structure of data records for each patient comprised: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), CGMS BG level (concentration), HR value (integer), Exercise (0/1)]. The records were organized chronologically, with the first record representing the earliest data recorded. Weight data was handled separately.

Table 3.4 presents the number of records extracted solely from the original D1namo dataset utilizing only CGMS data. Additionally, Table 3.5 displays the extracted data from the D1namo dataset incorporating both CGMS and HR features.

3.3.7 Explored Machine Learning Techniques

In this section, I outline the machine learning methodologies examined in this study along with their underlying technological principles. These techniques are predefined within the deployed Scikit-learn library [R81], facilitating easy parameterization. Through various tests conducted in my previous investigation within this domain [C5], I employed grid search based methods to ascertain the most suitable algorithm and its optimal configurations. Technical parameters for different methodologies were established based on my prior investigations, which involved the application of synthetic patient data derived from *in silico* experiments. As one of the objectives of my research was to assess the classification efficacy of models trained on synthetic data when applied to real patient data, I utilized the configurations previously determined for all models

Table 3.4: Properties of the cleaned data from the D1namo dataset used for patients' blood glucose feature extraction.

D1namo Patient	Number of records	Total time duration in days
Patient 1	939	6
Patient 2	1575	5
Patient 3	185	2
Patient 4	1383	4
Patient 5	1375	4
Patient 6	1225	6
Patient 7	966	5
Patient 8	1189	5
Patient 9	135	2
Total	8972	39

Table 3.5: Properties of the cleaned data from the D1namo dataset used for the extraction of patients' blood glucose and heart rate features.

D1namo Patient	Number of records	Total time duration in days
Patient 1	858	6
Patient 2	1440	5
Patient 3	166	2
Patient 4	1266	4
Patient 5	1251	4
Patient 6	1108	6
Patient 7	879	5
Patient 8	1088	5
Patient 9	85	2
Total	8141	39

and techniques in the experiments.

1. *Logistic Regression* (LR). The LR models provide the probability whether a given sample belong to a particular class or not by using the logistic function:

$$f(x) = M \cdot \exp[k(x - x_0)] , \quad (3.5)$$

where k is the steepness of the logistic curve, M is the maximum value of the curve and x_0 is the inflection point [R63]. During the training session I allowed maximum 1000 iterations and I applied L_2 -type penalty to measure the performance.

2. *AdaBoost Classifier* (AdaBoost) represents a boosting technique used in machine learning as an ensemble method. The weights are reassigned to each instance, with higher weights assigned to incorrectly classified instances. The purpose of boosting is to reduce bias and variances. It works on the principle of learners growing sequentially. Each subsequent learner is grown from previously grown learners except the first one. In simple words, weak learners are converted into strong ones [R64]. My model was set with maximum 50 trees.
3. *Decision Tree Classifier* (DC) [R65, R66]. The DC is a flowchart kind machine learning algorithm where each “node” represents a statistical probe on a given attribute. The “branches” represent the outcome of the probe while the leaf nodes represent a given class label. The paths from the roots to the leaves represent given classification rules. The goal of the DC is to learn rules by making predictions from features. Each branching point in the tree is a “rule” after which I get either the decision itself or a starting point of another subtree. The tree depth defines how many rules can be applied step by step to get the result. In this given case, I applied unlimited tree depth and all decisions are allowed to use any one of the features.
4. *Gaussian Naive Bayes* (Gaussian) [R67] is a very simple machine learning algorithm (also referred to as probabilistic classifier which is based on Bayes’ theorem). In general, the naive Bayes classifiers are highly scalable models, requiring a number of parameters linear in the number of variables in a given learning problem. When dealing with continuous or close-to-continuous data, a typical assumption is that the continuous values are associated with each class that are distributed according to a normal distribution.
5. *K-Nearest Neighbors Classifier* (KNN) [R68, R82]. The method is relied on labels and make an approximation function for a new data. The KNN algorithm assumes that the similar features “fall” close to each other (in numerical sense). The data is represented in a hyperspace based on the characteristics of the data. When new data arrives, the algorithm looks at the k nearest neighbors at the decision. The result of the decision is the class that received the most votes based on its neighbors. I use the $k - d$ tree implementation [R69] of KNN with $k = 5$.
6. *Support Vector Machines* [R70]. This machine learning algorithm can be used for classification, regression and also for outlier detection (anomaly detection). It is very efficient in high dimensional spaces, also good when the number of dimensions (features) is greater than the sample number. It is very versatile in case of using kernels. Common kernels are provided, but I can specify own kernels if I want. The main goal of the algorithm to is find a hyperplane in N -dimensional feature space. To separate classes I can find many different hyperplanes, but SVM finds the hyperplane which provides the maximum margin. The cost function is:

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y \cdot f(x) \geq 1 \\ 1 - y \cdot f(x), & \text{otherwise} \end{cases} , \quad (3.6)$$

where:

- $f(x)$ is the output of model,
- y the true output value,
- x the feature vector,
- c the cost function.

I examined SVMs with 1000 iterations in five kernel variants: radial basis function kernel (SVM kernel=rbf), sigmoid kernel (SVM kernel=sigmoid), 3rd degree polynomial kernel (SVM kernel=poly degree=3), 5th degree polynomial kernel (SVM kernel=poly degree=5) and 10th degree polynomial kernel (SVM kernel=poly degree=10).

7. *Random Forest* [R71, R72]. The Random Forest is based on decision trees: it creates different trees via training on different random sets of feature vectors from the training set that were selected according to the same rule. In prediction each tree gives a rating, a vote. These votes are aggregated. At the end of the summary, I will see which decision received the most votes. The decision that receives the most votes will be the final decision of the classifier. In my test the Random Forest was built with 100 trees (Random Forest);
8. *Multi Layer Perceptron Networks* (MLP) [R73, R83] A neuron consists of an input part, which is a vector, the weights of the neuron, which is also a vector, an activation function through which I pass the product of the input vector and the transposition of the weighting vector. The last element of the neuron is the output, which is the value of the activation function. Activation function can be sigmoid, tangent hyperbolic, Gaussian function, etc. An MLP is realized when multiple neurons are organized next to each other – that is the so-called layer – and multiple layers of neurons are arranged in a row that aggregate the input in a complex way to realized the output of the MLP. The output is obtained by going through the network layer by layer. The activation function is calculated for each neuron in each layer. In the end, the neuron with the highest value will be the output for that input, I.e. which neuron showed the highest activation for that input.

The activation function of the j th neuron from the I th layer is:

$$a_j^{(I)} = g(\theta_j^{(I-1) \cdot t} \cdot x) , \quad (3.7)$$

where x represents the feature vector, θ stands for the weights of the neuron, while g is the activation function (e.g. sigmoid).

The cost function is:

$$J(\theta) = -\frac{1}{m} \cdot \left[\sum_{I=1}^m \sum_{k=1}^K y_k^{(I)} \cdot \log(h_{\theta}(x^{(I)})_k) + (1 - y_k^{(I)} \cdot \log(1 - (h_{\theta}(x^{(I)}))_k)) \right] + \frac{\lambda}{2 \cdot m} \cdot \sum_{l=1}^L \sum_{I=1}^{s_l} \sum_{j=1}^{s_{l+1}} (\Theta_{ij}^{(l)})^2 \quad (3.8)$$

where

- m = number of samples
- K = number of neurons in actual layer
- L = number of layers

- s_l = error in actual layer
- s_{l+1} = error in next layer
- λ = regularization
- $h_\theta(x)$ = the output of actual layer.

In this work, I investigated different MLPs: four hidden layers of sizes 100, 150, 100, and 50, respectively, maximum 1000 iterations, and three variants of activation functions: logistic (MLP activation=logistic), ReLU (MLP activation=ReLU), and tanh (MLP activation=tanh).

3.3.8 Test Cases

1. Identifying physical activity solely from CGMS signals was achieved using the preprocessed data from the Ohio T1DM dataset, referenced as 1. The algorithms previously mentioned were thoroughly evaluated. My dataset was comprised of BG sample points, structured as follows: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), CGMS-derived BG level (concentration), Exercise indicator (0/1)]. The training dataset constituted 75% of the total data, with the remaining 25% reserved for testing purposes.
2. Identifying physical activity solely from CGMS signals was accomplished using the sanitized data from the D1namo dataset. The algorithms mentioned earlier were subjected to rigorous testing. My dataset was constructed with BG sample points, organized as follows: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), CGMS-derived BG level (concentration), Exercise indicator (0/1)]. For model training, 75% of the dataset was utilized, while the remaining 25% was held out for testing purposes.
3. Identifying physical activity based on both CGMS and HR signals was conducted using the sanitized data from the Ohio T1DM dataset, referenced as 2. The algorithms previously mentioned underwent comprehensive testing. My dataset was constructed to include both BG and HR sample points, organized as follows: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), CGMS-derived BG level (concentration), HR level (integer), Exercise indicator (0/1)]. For model training, 75% of the dataset was utilized, while the remaining 25% was reserved for testing purposes.
4. Identifying physical activity based on both CGMS and HR signals was carried out using the sanitized data from the D1namo dataset. The algorithms mentioned earlier were rigorously tested. My dataset incorporated both BG and HR sample points, structured as follows: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), CGMS-derived BG level (concentration), HR level (integer), Exercise indicator (0/1)]. For model training, 75% of the dataset was utilized, while the remaining 25% was reserved for testing purposes.
5. Identifying physical activity based on CGMS and HR signals was conducted using sanitized data from both the Ohio T1DM and D1namo datasets (for the Ohio T1DM dataset, I utilized the 2 dataset). The algorithms mentioned earlier underwent thorough testing. My dataset comprised BG sample points and was structured as follows: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), CGMS-derived BG level (concentration), HR level (integer), Exercise indicator (0/1)]. The training data consisted of records from the Ohio T1DM dataset, while the test data comprised records from the D1namo dataset. This use case can be applied to strengthen my hypothesis whether there are many non reported physical activities in the original Ohio T1DM dataset (where CGMS signal was available, physical activity was not reported, however, physical activity possible happened). Furthermore, this use-case can be applied for data annotation as well on the original Ohio T1DM dataset to select the presumable non-reported physical activities.

This scenario can serve to bolster my hypothesis regarding the potential presence of numerous unreported physical activities within the original Ohio T1DM dataset. In instances where CGMS signals were present but physical activity was not reported, this use-case explores the possibility that physical activity may have indeed occurred. Moreover, it can also be utilized for data annotation purposes within the original Ohio T1DM dataset, facilitating the identification and selection of presumable non-reported physical activities.

3.3.9 Feature extraction

Feature selection in this study was conducted to support predefined use-cases, specifically the recognition of physical activity from CGMS signals and from the CGMS and HR complex across different datasets. During the conceptualization phase, considerations were given to BG and HR variability dynamics. Previous research [C5] conducted various tests, indicating the beneficial nature of certain feature sets. I experimented with multiple models based on different algorithms sensitive to various aspects of the data, such as differences, velocities, and accelerations. Consequently, I opted to utilize the features introduced below, despite their correlated and overlapping information content.

I defined a window spanning 15 records to investigate BG variation, representing a 70-minute time window based on CGMS data provided every 5 minutes. Additionally, this window was subdivided into three "mini" windows to analyze BG level variations in a more detailed manner.

The *body weight of the patient* (w) was applied as "independent" feature.

The extracted features from the CGMS signal were the following:

- The *end-to-end difference of the blood glucose level* (d) in the window.

$$d = bg(14) - bg(0) \quad (3.9)$$

- The *blood glucose level difference between two consecutive sampled points* within the window (dp)

$$dp(i) = bg(i + 1) - bg(i), \quad \forall i = 0 \dots 13 \quad (3.10)$$

- The *change of the blood glucose levels in the three mini windows*, from beginning to end (dpp)

$$\begin{aligned} dpp(i) &= \sum_{j=0}^3 bg(5i + j + 1) - bg(5i + j) \\ &= bg(5i + 4) - bg(5i) \end{aligned} \quad (3.11)$$

for any $i = 0 \dots 2$.

- *End-to-end blood glucose level change speed* (v):

$$v = \frac{bg(14) - bg(0)}{t(14) - t(0)} = \frac{d}{14 \cdot \Delta t} \quad (3.12)$$

- The *blood glucose level change speed between two consecutive sampled points* (vp)

$$vp(i) = \frac{bg(i + 1) - bg(i)}{t(i + 1) - t(i)} = \frac{dp(i)}{\Delta t}, \quad \forall i = 0 \dots 13 \quad (3.13)$$

- The *blood glucose level change speed in all mini sliding window measured points*. From beginning to end (*vpp*)

$$vpp(i) = \frac{\sum_{j=0}^3 (bg(5i+j+1) - bg(5i+j))}{t(5i+4) - t(5i)} = \frac{dpp(i)}{4 \cdot \Delta t}, \quad (3.14)$$

for any $i = 0 \dots 2$.

- The *acceleration of blood glucose level change speed among three consecutive sampled points* (*ap*)

$$ap(i) = \frac{bg(i+2) - bg(i)}{(t(i+2) - t(i))^2}, \quad i = 0 \dots 12. \quad (3.15)$$

The extracted features from the HR signal were the following:

- The *HR measured at the CGMS sample time* (*hr*),

$$hr(i) \quad (3.16)$$

- The *end-to-end heart rate difference*, the difference between the first heart rate measured point and the last heart rate measured point, which points is between two consecutive glucose sample times (*hrp*),

$$hrp = hr(i+1) - hr(i) \quad (3.17)$$

Because of the feature selection, particularly due to parameter d , the developed models require 15 BG registrations to operate effectively. To circumvent the gap during model initiation, I fill in the missing data with 0 values while the database is "loading." Consequently, the models function, but precise estimations are only possible after 15 BG registrations, which translates to a 75-minute delay from the initiation, considering a 5-minute sampling time in the CGMS signal.

Although I did not consider chronological order as a feature, I maintained the real-time sequence of data recordings. The input to the models consisted of defined features extracted from the cleaned data. For CGMS recordings only, I applied the following feature set: $FS_1 = [w, d, dp(0), dp(1), dp(2), dp(3), dp(4), dp(5), dp(6), dp(7), dp(8), dp(9), dp(10), dp(11), dp(12), dp(13), dpp(0), dpp(1), dpp(2), v, vp(0), vp(1), vp(2), vp(3), vp(4), vp(5), vp(6), vp(7), vp(8), vp(9), vp(10), vp(11), vp(12), vp(13), vpp(0), vpp(1), vpp(2), ap(0), ap(1), ap(2), ap(3), ap(4), ap(5), ap(6), ap(7), ap(8), ap(9), ap(10), ap(11), ap(12)]$. When both CGMS and HR recordings were considered, I utilized the following feature set: $FS_2 = [w, d, dp(0), dp(1), dp(2), dp(3), dp(4), dp(5), dp(6), dp(7), dp(8), dp(9), dp(10), dp(11), dp(12), dp(13), dpp(0), dpp(1), dpp(2), v, vp(0), vp(1), vp(2), vp(3), vp(4), vp(5), vp(6), vp(7), vp(8), vp(9), vp(10), vp(11), vp(12), vp(13), vpp(0), vpp(1), vpp(2), ap(0), ap(1), ap(2), ap(3), ap(4), ap(5), ap(6), ap(7), ap(8), ap(9), ap(10), ap(11), ap(12), hr, hrp]$. These feature sets, FS_1 and FS_2 , were utilized as inputs during model operation.

Figure 3.5 illustrates the graphical representation of the feature extraction process described above.

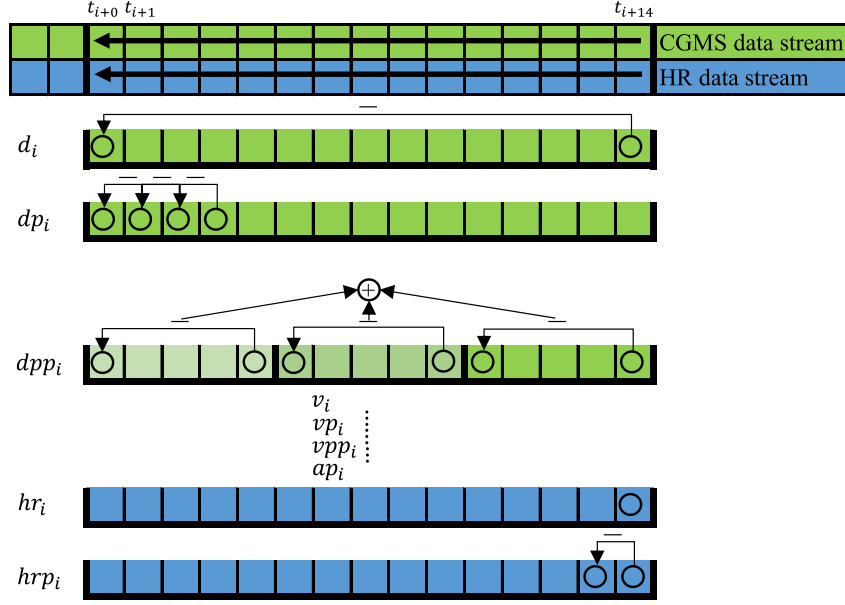


Figure 3.5: Abstraction of feature extraction during operation. The v , vp , vpp and ap features are originated from the d kind features, thus these are not listed here. In the case of the dp_i only the first four value has represented as a demonstration, however, all sampled values has been considered from the window during operation.

3.3.10 Results Real Patient Dataset

In this section, I present the outcomes obtained when utilizing both CGMS and HR features from two datasets. However, the training dataset employed was Ohio T1DM, while the test dataset was D1namo. Essentially, the models underwent training solely on the Ohio T1DM dataset, with subsequent testing conducted on the D1namo dataset. This approach serves as a robust test of the research hypothesis, with potential to address two key questions: first, whether the models demonstrate robustness according to the specified metrics, and second, whether the models can effectively annotate the Ohio T1DM dataset with satisfactory performance. Figure 3.6 illustrates the test results. Notably, both LR and AdaBoost methods yielded superior AUC scores (surpassing 0.9), while Random Forest and two MLP models also achieved commendable AUC-related performance. Table 3.6 outlines the performance metrics of the models. Notably, five models attained an AUC of ≥ 0.8 : Logistic Regression, AdaBoost, Random Forest, MLP with ReLU activation function, and MLP with Tanh activation function. Consequently, the subsequent evaluation centers on these models. Logistic Regression achieved an accuracy (ACC) of 0.845, the highest among the models. Of particular significance, the True Positive Rate (TPR) attained a value of 0.818, indicating the model's capability to predict nearly 82% of positive cases accurately. Moreover, the Positive Predictive Value (PPV) yielded a result of 0.864, implying that the LR model makes erroneous decisions in only 14% of cases when predicting positive. Remarkably, LR also boasts the highest PPV value among all models. The LR model exhibited the highest F1 score at 0.844, though slightly lower than the 0.9 achieved in the prior use-case. AdaBoost demonstrated a notable TPR score of 0.909, surpassing all other models in the test. Consequently, AdaBoost effectively identifies positive classes but occasionally misclassifies negative ones. Its PPV value of 0.80 falls 6% below LR's precision. The F1 score for AdaBoost was 0.838, demonstrating performance close to LR. Remarkably, the Random Forest model coincidentally attained identical scores of 0.818 across TPR, ACC, PPV, and F1 metrics. This suggests the model accurately identifies 82% of physical activity while making an 18% error rate in predicting the negative class. Contrary to expectations, the MLP variants, regardless of the tested activation functions, did not yield predictions as precise as the aforementioned

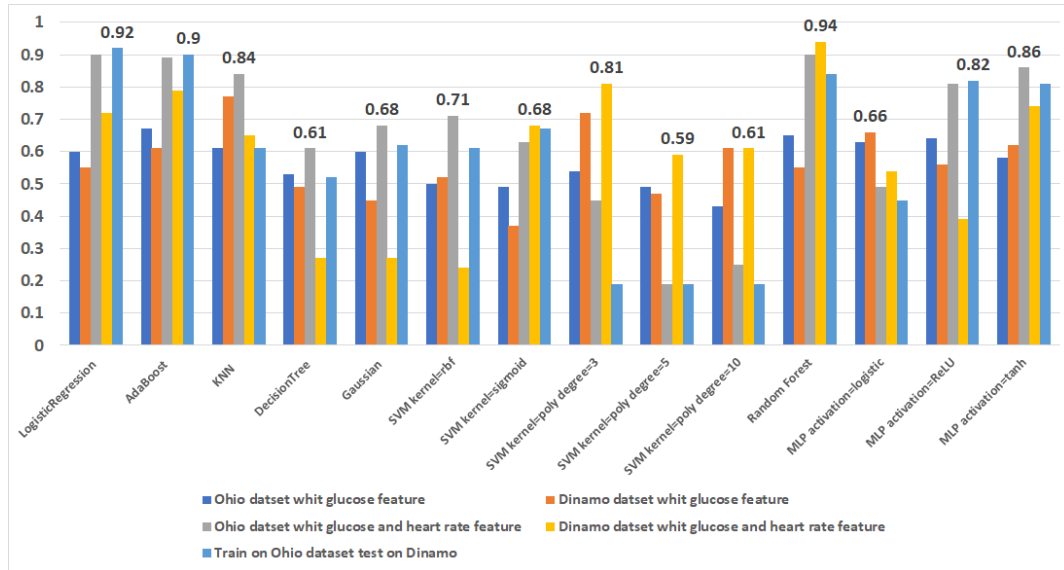


Figure 3.6: AUC of models in all use-cases – test results

Table 3.6: AUC performance achieved in different tests

Training dataset	Ohio T1DM	D1namo	Ohio T1DM	D1namo	Ohio T1DM
Testing dataset	Ohio T1DM	D1namo	Ohio T1DM	D1namo	D1namo
Features	BG only	BG only	BG and HR	BG and HR	BG and HR
Logistic Regression	0.60	0.55	0.90	0.72	0.92
AdaBoost	0.67	0.61	0.89	0.79	0.90
KNN	0.61	0.77	0.84	0.65	0.61
Decision Tree	0.53	0.49	0.61	0.27	0.52
Gaussian	0.60	0.45	0.68	0.27	0.62
SVM kernel=rbf	0.50	0.52	0.71	0.24	0.61
SVM kernel=sigmoid	0.49	0.37	0.63	0.68	0.67
SVM kernel=poly, deg=3	0.54	0.72	0.45	0.81	0.19
SVM kernel=poly, deg=5	0.49	0.47	0.19	0.59	0.19
SVM kernel=poly, deg=10	0.43	0.61	0.25	0.61	0.19
Random Forest	0.65	0.55	0.90	0.94	0.84
MLP activation=logistic	0.63	0.66	0.49	0.54	0.45
MLP activation=ReLU	0.64	0.56	0.81	0.39	0.82
MLP activation=tanh	0.58	0.62	0.86	0.74	0.81

algorithms.

Overall, the results indicate AdaBoost as the "best" algorithm for this particular test case.

3.3.11 OHIO T1DM Dataset for Recurrent Neural network

The OHIO T1DM dataset is a resource designed for researchers aiming to enhance the health and management of individuals with type 1 diabetes. It includes data collected over an 8-week period from 12 participants with type 1 diabetes.

This dataset features several types of information, including continuous glucose monitoring (CGM) data with glucose levels recorded every 5 minutes. It also contains blood glucose measurements obtained through periodic self monitoring with fingersticks. In addition, the dataset includes details on insulin doses (both bolus and basal), self reported meal times with carbohydrate estimates, and personal information about exercise, sleep, work, stress, and illness. It also features physiological data from fitness bands and environmental information.

To ensure participant privacy, individuals in the dataset are anonymized and identified only

Table 3.7: Glucose hours by patients

540	544	552	559	563	567	570	575	584	588	591	596
1242	1115	960	1115	1228	1112	1148	1213	224	1290	1138	1140

by unique identifiers. The OHIO T1DM dataset was first made available during the Blood Glucose Level Prediction (BGLP) Challenge in 2018 and 2020. [R84].

To achieve my research goals, I employed a multifaceted approach that involved using three distinct methodologies. I systematically extracted relevant data types from the dataset to support my investigations. This data includes critical physiological parameters such as glucose levels, heart rate, and step counts.

The glucose level data consists of continuous glucose monitoring (CGM) measurements taken every five minutes. Heart rate data, also collected in five-minute increments, is available only for participants who used the Basis Peak sensor band. Similarly, step count data, aggregated at five-minute intervals, is restricted to those who wore the Basis Peak sensor band.

- `glucose_level`: The glucose data comprises continuous glucose monitoring (CGM) measurements in milligrams per deciliter (mg/dl), with corresponding timestamps recorded at five-minute intervals. The timestamp format follows the DD-MM-YYYY HH:MM:SS pattern.
- `basis_heart_rate`: Heart rate information is aggregated in five-minute increments and is exclusively accessible for individuals who utilized the Basis Peak sensor band. Heart rate recordings include timestamps, denoting the date and time (in DD-MM-YYYY HH:MM:SS format), along with corresponding heart rate data measured at five-minute intervals (in beats per minute).
- `basis_steps`: The dataset comprises step counts aggregated every 5 minutes in the DD-MM-YYYY HH:MM:SS format. This data is also exclusively accessible for individuals using the Basis Peak sensor band.

Table 3.7 summarizes the glucose data, listing the patient analyzed in the first row and the duration of glucose data collection in hours in the second row.

During the initial phase of data preprocessing, I meticulously refined the original dataset using specific criteria. The primary focus was on identifying and addressing missing values, especially within the heart rate and step data fields. I carefully examined these parameters to ensure that no missing entries were present.

If any missing values were detected in heart rate or step data, the entire corresponding row was excluded from the CSV dataset.

Additionally, I conducted a temporal analysis of the glucose measurements to identify and address any temporal gaps. I closely examined the intervals between consecutive glucose measurements. If the time between any two measurements exceeded a predefined threshold of five minutes, the dataset was reorganized. This involved segmenting the dataset at each instance of such temporal discontinuity and treating each segment as a separate dataset.

This approach aimed to maintain the temporal integrity of the data and enhance the accuracy of subsequent analyses.

As described earlier, my research employed a tripartite methodological framework, which required the development of three distinct dataset structures:

1. The first dataset structure exclusively comprised glucose data. This univariate configuration allowed for an in-depth analysis of glucose dynamics, unencumbered by the influence of additional physiological variables.

In this case, the data record for each patient in this dataset had the following structure: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), Blood glucose level from CGMS (concentration)].

2. In the second dataset structure, my analytical scope expanded to encompass the dynamic interplay between glucose levels and heart rate. This bivariate approach facilitated a more nuanced examination by integrating heart rate data, also aggregated at five-minute intervals. Importantly, this dataset structure was specifically tailored for individuals who wore the Basis Peak sensor band, ensuring methodological consistency and uniformity in data acquisition practices.

In this case, the data record for each patient in this dataset had the following structure: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), Blood glucose level from CGMS (concentration), HR value (integer)].

3. The third dataset structure extended the integrative paradigm by pairing glucose data with step information. Similar to the previous approach, the aggregation of data occurred at five-minute intervals, and exclusivity was maintained for individuals employing the Basis Peak sensor band.

In this case, the data record for each patient in this dataset had the following structure: [Date stamp (DD-MM-YYYY), Time stamp (HH:MM:SS), Blood glucose level from CGMS (concentration), Step value (integer)].

Essentially, the creation of these three distinct dataset structures represents a thoughtful and strategic research approach. By systematically varying the combinations of physiological parameters, the objective was to reveal patterns and relationships within the data. This method contributes to a deeper understanding of the intricate interactions between glucose levels, heart rate, and step count.

3.3.12 Recurrent neural network approach

In this study [J2], I focused on general machine learning algorithms, particularly recurrent neural networks (RNNs), which are well-suited for time-series data such as physical activity detection. Given the uniform time intervals in my dataset, RNNs are an appropriate choice and offer a modern alternative to traditional machine learning algorithms.

The architectural designs of my models were similar, with the main differences lying in the recurrent layers. Specifically, I compared networks using Long Short Term Memory (LSTM) cells and Gated Recurrent Units (GRU) cells. I also explored variations in other parameters, such as the look back time horizon, which ranged from 3 to 24 time steps, equivalent to 15 minutes to 2 hours given my 5-minute interval data.

Additionally, I considered the size of the feature vectors in the input layer, which is determined by both the time horizon and the types of sensor data used. My dataset includes data from blood glucose meters, heart rate monitors, and step counters. To address overfitting, I adjusted the dropout rate between 0, 0.2, and 0.5 across all layers.

Another important parameter was the number of RNN cells, which ranged from 16 to 128 and was consistent across all recurrent layers. The dense layer, responsible for the hidden layer neurons, had configurations of 64, 128, 256, 512, and 1024 neurons.

These parameters defined my network configurations, and each configuration was systematically trained and tested to evaluate performance.

Proposed model

The architecture of the LSTM model is illustrated in Figure 3.7 (right panel). The model's input layer is influenced by two factors: the number of time steps considered and the number of

features used. In this case, I looked back 24 steps, equivalent to two hours of data, and used 2 features from two sensors.

Following the input layer is a Bidirectional layer [R85] [R86] that incorporates LSTM cells in both the forward and backward directions. The number of LSTM cells in this layer was a variable parameter, set to 128 for the network depicted in the figure. The RNN layer has the `return_sequences` property set to `true`, meaning it provides an output at every time step rather than just the final step. Additionally, a dropout rate was applied to this layer to mitigate overfitting.

Next, a Batch Normalization [R87] layer normalizes the data. This is followed by another Bidirectional layer with identical parameters to the first one, including the same number of RNN cells and the `return_sequences` setting. A subsequent Batch Normalization layer is added, and then a final Bidirectional layer is incorporated, mirroring the structure of the previous two Bidirectional layers with consistent RNN cell numbers, `return_sequences` parameters, and dropout rates.

After this, a final Batch Normalization layer is introduced to normalize the data once more. The next component is a Global Average Pooling [R88] layer, which reduces multiple time vectors into a single vector by averaging. The output from this layer is a vector with a length equal to the number of RNN cells.

Following the pooling layer, the first Dense layer is configured with a neuron count set according to the input parameter, in this case, 256. This is followed by a Dropout layer with a rate matching that of the RNN layer to prevent overfitting. The second Dense layer also contains the same number of neurons as the first Dense layer. Another Dropout layer is added, maintaining consistency with other dropout rates in the network. ReLU [R89] activation functions are used in the Dense layers.

Finally, the classification layer consists of two neurons to represent the two possible states, with a softmax [R90] activation function. For optimization, the Adam optimizer [R91] is employed, and sparse categorical crossentropy [R92] is used as the cost function.

Let's start by outlining the architecture of the GRU model, as depicted in Figure 3.7. The model's input depends on two factors: the number of time steps I look back (24 steps, or two hours of data) and the number of features (2 features from two sensors).

The input layer is followed by a Bidirectional layer, which is typical in the GRU model. This layer contains GRU cells operating in both forward and backward directions. The number of GRU cells, set to 128 in the depicted network, is a variable parameter.

Following this is an RNN layer with the `return_sequences` property set to `true`, meaning it outputs a value at each time step rather than just the final one. A dropout rate is applied to this layer to prevent overfitting. Next, a Batch Normalization layer normalizes the data.

Another Bidirectional layer, with parameters identical to the first (including the same number of GRU cells, `return_sequences` setting, and dropout rate), is added. This is followed by another Batch Normalization layer.

The final Bidirectional layer mirrors the configuration of the previous two layers, maintaining consistent GRU cell numbers, `return_sequences` parameters, and dropout rates. After this, a final Batch Normalization layer ensures data normalization.

A Global Average Pooling layer then consolidates the multiple time vectors into a single vector by averaging. This output vector has a length equal to the number of GRU cells.

The first Dense layer is configured with a neuron count specified by the input parameter, set to 256 in this case. Following this, a Dropout layer is used to combat overfitting, with the dropout rate matching that of the RNN layer. The second Dense layer also has a neuron count of 256, and is accompanied by another Dropout layer with consistent dropout values. ReLU activation functions are applied in the Dense layers.

Finally, the classification layer consists of two neurons to represent the two possible states, and employs a softmax activation function. The model uses the Adam optimizer, with sparse

categorical crossentropy as the cost function. Overall, these choices for optimization, activation, and loss functions are well suited and effective.

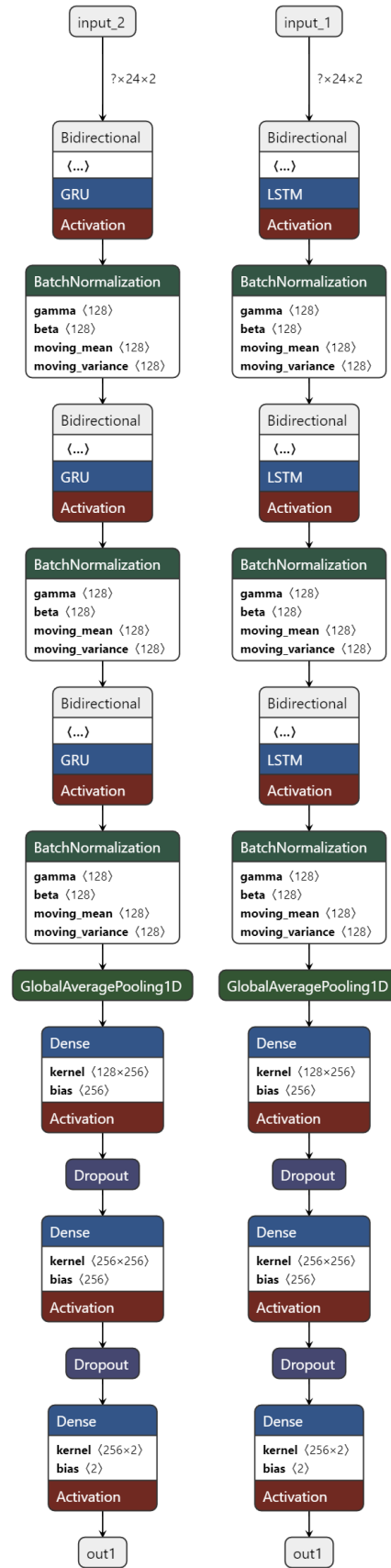


Figure 3.7: The structure of the used GRU and LSTM models. All model configurations had the same structure, only the values of the hyper parameters changed. The kernel is represented in the image using a matrix of how much data it processes.

Training and testing

Eighty percent of the dataset was allocated for training, while the remaining twenty percent was used for testing. Given the importance of maintaining temporal order in time series data, the split was carefully managed to ensure that data points were consecutive in time. Additionally, efforts were made to minimize overlap between the training and testing datasets. To further validate the model, cross validation was employed during training. For each set of parameters, five training and testing iterations were conducted. In these iterations, the testing dataset was initially the first 20% of the data, and by the fifth iteration, it was the last 20%. The remaining data was used for training. Each training session consisted of 1000 epochs, with a batch size of 256. Additionally, the model was saved whenever there was a decrease in the cost function value on the train dataset. During the testing phase, the model with the lowest cost function value was then selected. This approach was necessary because of the class imbalance, which made accuracy an unreliable metric for evaluation in this context.

3.3.13 Result with Recurrent Neural Network

Next, the results of the models are reviewed to determine which parameter configurations are sufficient to meet the performance requirements. To achieve this, performance metrics for various parameter configurations were collected and visualized using box plots. The metrics analyzed include Accuracy, Precision, Recall, and F1 score.

For both GRU and LSTM models, the top 30 models were evaluated based on F1 score, and the corresponding box plots for Accuracy, Precision, and Recall metrics are provided and detailed in the supplementary material. Additionally, two tables in the supplementary material offer further insights: one table presents the AUC and Precision values of the top 30 models, while the other displays the Precision and Recall values for these models.

F1 score

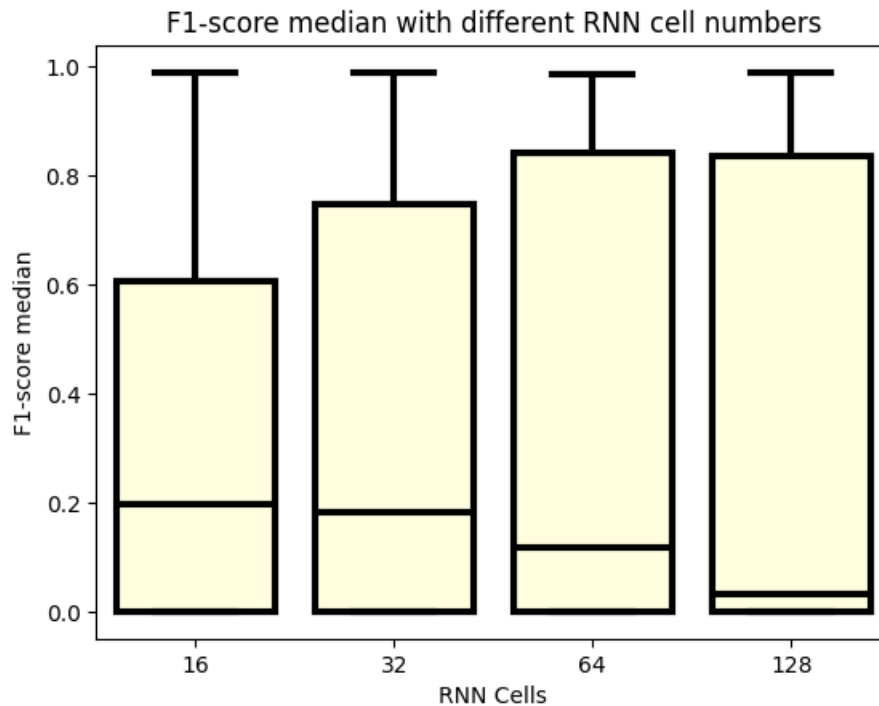


Figure 3.8: F1 score value for different number of RNN cell

In Figure 3.8, the F1 score values are analyzed in relation to the number of RNN cells. A clear trend is observed: higher numbers of RNN cells generally correspond to higher upper quartile values, suggesting that 25% of the models with more cells perform better. However, the median values show an inverse trend, with the lowest medians recorded for the largest cell numbers, specifically 64 and 128.

According to the F1 score analysis, models with either 64 or 128 RNN cells achieve the best results. Nonetheless, configurations with 16 cells can also achieve F1 scores close to 1. On average, models with 64 or 128 RNN cells tend to deliver the best performance.

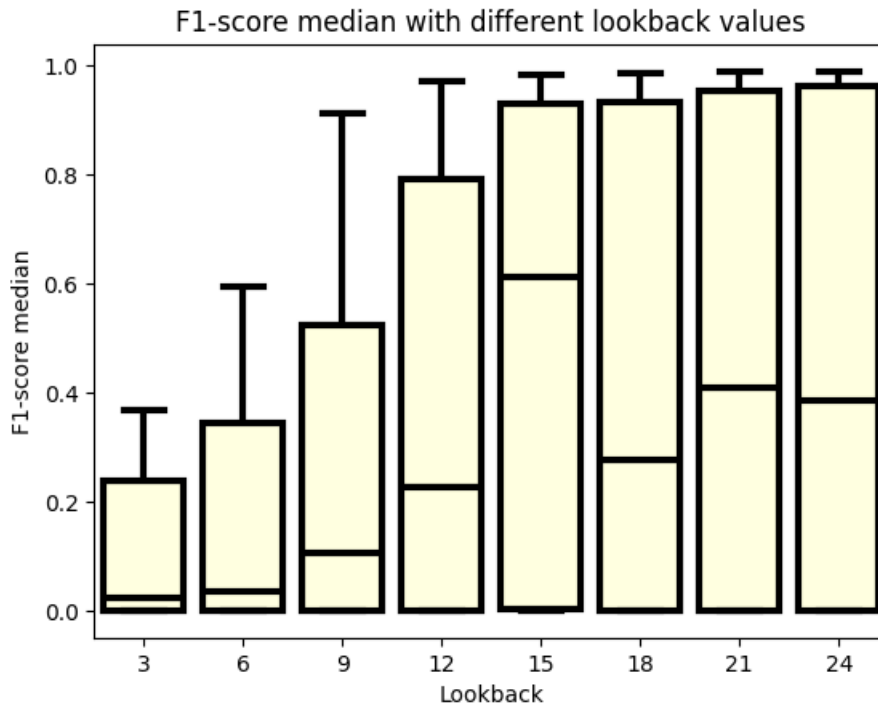


Figure 3.9: F1 score value for different number of Look back

In Figure 3.9, the F1 score is analyzed in relation to various look-back window values. A staircase pattern similar to that observed for Precision and Recall is evident, as the F1 score combines these two metrics. The median F1 scores show a steady increase up to a 15-fold look-back window. Notably, some models perform well even with a 12-fold look-back. However, it is at the 15-fold look-back window where the upper quartile of the F1 score exceeds 0.8. Before this, only the maximum value of the boxplot, seen for 9-fold and 12-fold look-backs, reaches this level.

For look-back windows greater than 15, while the median F1 scores are slightly lower than for the 15-fold look-back, the upper quartile values are higher. Specifically, at a 24-fold look-back, the top 25% of models perform better compared to those with a 15-fold look-back. However, the lower median scores for larger look-back windows can be attributed to the vanishing gradient problem. Models that manage to avoid this issue may outperform those with a 15-fold look-back, but when the problem occurs, performance can deteriorate significantly. A 15-fold look-back window achieves good performance on average, whereas a 24-fold look-back window yields the best results.

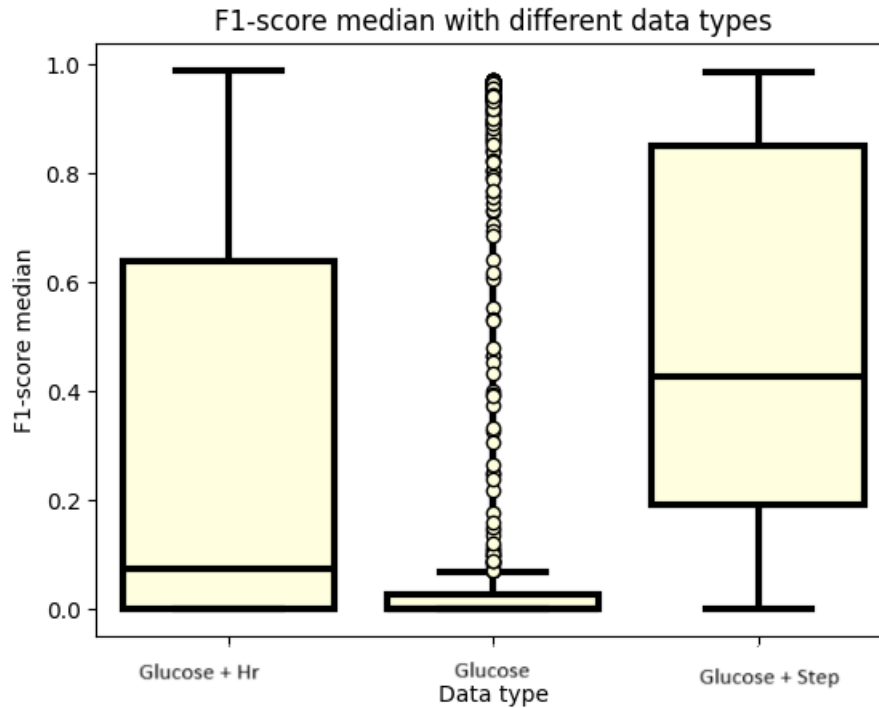


Figure 3.10: F1 score value for different datatype

Figure 3.10 displays the F1 score values across different datasets. Outliers are particularly notable when only blood glucose values are used as features. While some outlier models achieve F1 scores close to 1, such instances are infrequent. When blood glucose and heart rate data are used together, the median F1 score remains below 0.2. However, the upper quartile surpasses 0.6, and the maximum F1 score approaches 1. Utilizing both blood glucose and step count data as features leads to a significant improvement in performance. In this scenario, the median F1 score is around 0.6, with the lower quartile exceeding 0.8, demonstrating the enhanced effectiveness of incorporating both blood glucose and step count data. Despite this, it is important to note that some models perform well using only blood glucose data.

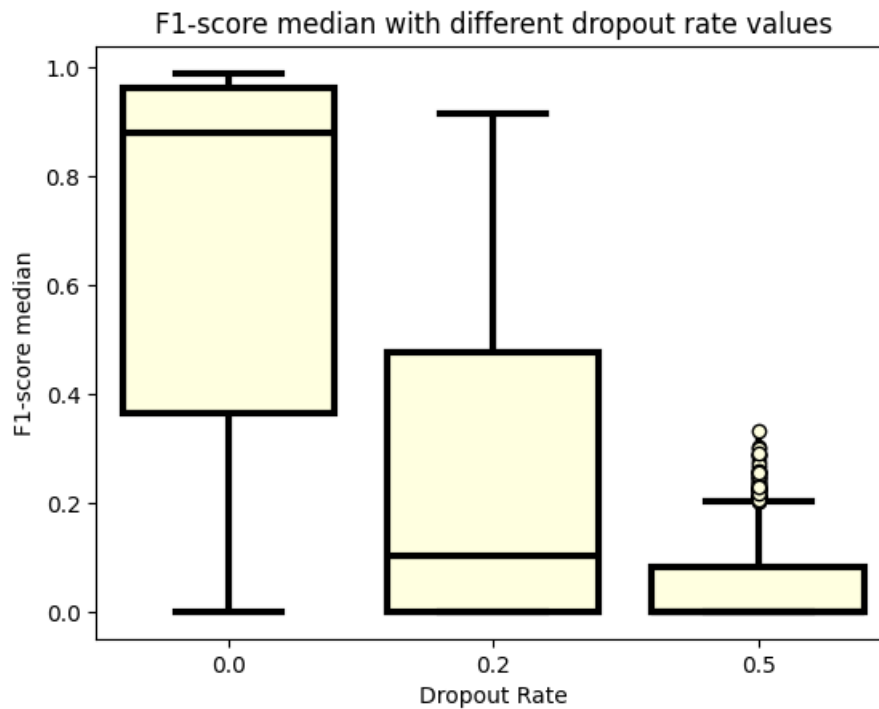


Figure 3.11: F1score value for different number of Drop out rate

Figure 3.11 illustrates the F1 score values for different dropout rates. The boxplots reveal that incorporating dropout rates into the model design may not be advantageous. Even a small dropout rate of 0.2 results in significant performance degradation, suggesting that models have difficulty generalizing to the data. This issue is further intensified with a dropout rate of 0.5, which leads to the poorest F1 scores. In contrast, the complete omission of dropout results in better performance, with median F1 scores approaching 0.8 and maximum values nearing 1. Thus, it might be beneficial to avoid using dropout rates in model design to achieve optimal performance.

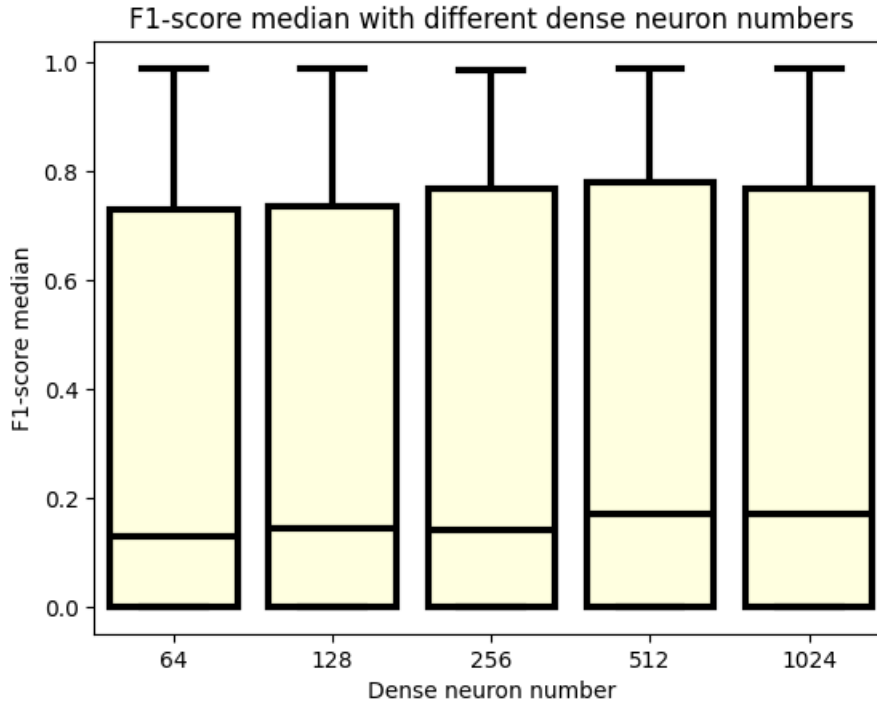


Figure 3.12: F1 score value for different number of Dense neurons

Figure 3.12 presents the F1 score values, indicating that the number of neurons in the dense layer has a minimal impact on model performance. Although the median F1 scores are somewhat higher for configurations with 512 and 1024 neurons, the difference is not significant. Notably, some models with the smallest neuron count of 64 also achieve very good performance, suggesting that this setup can still be effective. Nevertheless, neuron counts of 256, 512, and 1024 appear to provide slight advantages, as indicated by higher upper quartile values relative to the 64 and 128 neuron configurations. This suggests that the top 25% of models may perform slightly better with these larger neuron counts, though the improvement is marginal.

Analysis of the best 30 models

In this subsection, I rank the top 30 models based on their F1 scores, as this metric effectively balances precision and recall, ensuring that the models demonstrate strong performance in both aspects of classification accuracy. Due to the extensive nature of the data, I have divided the results into three tables. The ranking is based on the median F1 score from five test cases. Table 3.8 presents these rankings and F1 score values. Additional tables in the supplementary material provide further details: one table shows the AUC and accuracy results, while another displays the precision and recall values for the tested models.

Table 3.8 presents the F1 score values, which are arguably the most critical metric. A review of the median F1 scores reveals that all models in my dataset consistently achieve scores above 0.98. When examining the mean scores, only one configuration among the top thirty models falls below this threshold. Additionally, the variance among the models is remarkably low, highlighting the robustness and reliability of my results.

The analysis of my extensive dataset, comprising over 3,000 test cases, demonstrates a strong solution to the problem of physical activity detection and offers insights into the parameters affecting this process. By examining boxplots of various parameter settings, I assessed which factors influenced the results positively. Compared to previous studies that achieved an AUC of 0.92 with simpler machine learning algorithms, my use of recurrent layers has improved performance to an AUC of 0.99, with a F1 score of 0.98.

Model	Data Type	Look back	Dropout Rate	RNN Cells	Dense Neuron Number	F1 Score		
						Mean	Median	STD
LSTM	Glucose and HR	24.0000	0.0000	32.0000	64.0000	0.9843	0.9884	0.0063
LSTM	Glucose and HR	21.0000	0.0000	128.0000	128.0000	0.9876	0.9875	0.0021
LSTM	Glucose and HR	24.0000	0.0000	16.0000	1024.0000	0.9839	0.9869	0.0068
LSTM	Glucose and HR	24.0000	0.0000	128.0000	512.0000	0.9847	0.9869	0.0052
GRU	Glucose and HR	24.0000	0.0000	128.0000	1024.0000	0.9873	0.9860	0.0054
LSTM	Glucose and HR	24.0000	0.0000	64.0000	64.0000	0.9835	0.9858	0.0040
GRU	Glucose and HR	24.0000	0.0000	128.0000	256.0000	0.9840	0.9852	0.0045
LSTM	Glucose and HR	21.0000	0.0000	128.0000	64.0000	0.9852	0.9849	0.0012
LSTM	Glucose and HR	24.0000	0.0000	128.0000	64.0000	0.9856	0.9848	0.0044
GRU	Glucose and HR	24.0000	0.0000	128.0000	64.0000	0.9854	0.9848	0.0033
LSTM	Glucose and HR	21.0000	0.0000	128.0000	1024.0000	0.9840	0.9848	0.0051
LSTM	Glucose and HR	24.0000	0.0000	64.0000	128.0000	0.9820	0.9845	0.0051
LSTM	Glucose and Steps	24.0000	0.0000	128.0000	128.0000	0.9839	0.9844	0.0038
LSTM	Glucose and HR	24.0000	0.0000	16.0000	256.0000	0.9821	0.9843	0.0051
LSTM	Glucose and HR	18.0000	0.0000	128.0000	256.0000	0.9843	0.9842	0.0012
LSTM	Glucose and HR	24.0000	0.0000	128.0000	128.0000	0.9840	0.9841	0.0028
LSTM	Glucose and Steps	24.0000	0.0000	128.0000	256.0000	0.9835	0.9841	0.0017
GRU	Glucose and HR	24.0000	0.0000	128.0000	128.0000	0.9838	0.9841	0.0031
GRU	Glucose and Steps	24.0000	0.0000	128.0000	128.0000	0.9835	0.9840	0.0053
LSTM	Glucose and HR	24.0000	0.0000	64.0000	256.0000	0.9827	0.9837	0.0046
LSTM	Glucose and HR	24.0000	0.0000	128.0000	256.0000	0.9839	0.9836	0.0028
LSTM	Glucose and HR	24.0000	0.0000	32.0000	256.0000	0.9800	0.9835	0.0081
LSTM	Glucose and HR	24.0000	0.0000	32.0000	512.0000	0.9851	0.9834	0.0048
LSTM	Glucose and Steps	24.0000	0.0000	128.0000	64.0000	0.9833	0.9833	0.0025
GRU	Glucose and HR	24.0000	0.0000	64.0000	256.0000	0.9824	0.9833	0.0024
LSTM	Glucose and HR	21.0000	0.0000	128.0000	512.0000	0.9786	0.9833	0.0101
GRU	Glucose and Steps	24.0000	0.0000	128.0000	64.0000	0.9823	0.9832	0.0040
GRU	Glucose and HR	24.0000	0.0000	64.0000	512.0000	0.9815	0.9831	0.0040
GRU	Glucose and HR	21.0000	0.0000	64.0000	1024.0000	0.9788	0.9830	0.0097
LSTM	Glucose and HR	24.0000	0.0000	32.0000	128.0000	0.9826	0.9830	0.0038

Table 3.8: The 30 best model F1 scores

The results highlight that blood glucose levels alone are insufficient for achieving high classification performance. While there are instances where blood glucose levels can yield good results, this is not consistent across all cases. Blood glucose levels showed the best performance in some datasets, but outlier values indicate that such performance is not generalizable. Combining blood glucose with heart rate data yielded mixed results: although the maximum values were good, the upper quartile also showed promise. However, my findings suggest that using heart rate data alone is not ideal for detecting physical activity; cadence data is more effective in capturing the onset of activity, as it avoids the issues that arise with heart rate measurements during stressful situations.

Regarding dropout rates, my experiments confirm that while dropout can help prevent overfitting, excessively high rates are detrimental. A dropout rate of 0.2 remains a viable option, but a rate of 0.5 was found to be less effective. The best results were achieved when no dropout rate was applied to either recurrent or dense layers.

The look-back window parameter also significantly affects model performance. Models perform best with a minimum window of 15 steps (or approximately an hour). While extending the window beyond 15 steps provides minimal additional improvement, having at least an hour of data is crucial. The change in RNN cell counts did not significantly impact model performance, but models with higher cell counts generally performed better. A 64-cell configuration is recommended, though the best model had 32 cells, despite its high variance.

Lastly, the number of neurons in the dense layer had the least impact on model performance. Despite variations, the top models showed similar results across different neuron counts. Compared to my previous work [J1], my models have shown significant progress. In [R93], various machine learning algorithms achieved a maximum AUC of 0.92, while my LSTM model reached an F1 score of 0.98, demonstrating notable advancement.

3.3.14 Conclusion

In conclusion, I successfully met my objectives by achieving a higher F1 score of 0.9, surpassing the performance of my previous research. My earlier work, which utilized simpler machine learning algorithms, reached a maximum AUC of 0.92. In contrast, my current experiments have attained an AUC of 0.99, demonstrating a significant advancement. The fact that multiple recurrent models achieved this high AUC further confirms that recurrent architectures are effective for physical activity detection.

My research also explored various parameters impacting model performance. It was evident that using blood glucose levels alone is insufficient for building robust models. Although there were exceptional cases where blood glucose alone was adequate, these instances were outliers. A more reliable approach involves combining blood glucose with cadence data, rather than heart rate, which proved less effective.

I investigated how different look-back window sizes influence model performance and found that models require more than an hour of data to perform well. A look-back window of at least 15 steps is recommended, though extending beyond 15 steps provides only marginal improvement.

Regarding dropout rates, my results indicate that while a dropout rate of 0.2 can yield good results, it is an exception rather than the norm. Rates of 0.5, however, result in poorer performance. Therefore, it is advisable to use no dropout rate for optimal results.

The analysis of RNN cell sizes confirmed that exceeding 64 cells is unnecessary. Additionally, the number of neurons in the dense layer was found to have minimal impact on model performance, although it affects runtime.

No significant difference was observed between GRU and LSTM models in my study. For future research, exploring different data representations that include blood glucose, heart rate, and step count together could be valuable. Investigating the results from combining heart rate and step count, as well as experimenting with transformer models, could provide further

insights. Additionally, conducting tests where the training dataset remains the OHIO dataset but the testing dataset consists of new, measured data would be worthwhile.

3.4 Gesture detection

In this chapter, I will describe a research project that facilitates automatic detection of meal intakes. Using data from smart devices, it can help diary keeping in diabetes patients.

3.4.1 Goal

This research focused on gesture detection, specifically identifying eating gestures. The goal was to develop an AI system capable of classifying whether a gesture represents a meal, using raw data collected from smart devices. This is particularly important for elderly patients, who often forget to log their carbohydrate intake, making it difficult for doctors to properly adjust their insulin treatment. Therefore, having an AI that can autonomously detect carbohydrate intake is crucial. Initially, I based the study on accelerometer data, but it became evident that this alone was insufficient.

3.4.2 Introduction

Food consumption is a fundamental human activity, closely tied to physiological health particularly in individuals with diabetes [R94] [R95] [R96] [R97] [R98]. Accurately identifying and understanding food intake is essential for effective diabetes management, as it plays a key role in a holistic care strategy. This, in turn, contributes to better blood glucose regulation, more informed dietary decisions, and empowers patients to take greater control of their condition [R99], [R100], [R101]. With diabetes treatment becoming a global concern, monitoring food intake and physical activity is essential. However, traditional methods like journaling often lack accuracy due to manual entry and reliance on memory [R102]. The use of automated food consumption detection through wearable sensor systems is gaining attention as a promising solution [R103]. Recent studies indicate that deep learning-based gesture recognition, utilizing accelerometer and gyroscope data, outperforms traditional machine learning methods [R104]. Traditional methods of detecting food consumption rely heavily on self-reported data, which is prone to human error and can be a time-consuming manual process. Automated food consumption detection offers a solution by independently collecting and processing data, complementing traditional methods and reducing bias. These systems also offer the potential for personal self-monitoring by providing tailored food recommendations [R105].

Beyond its basic nutritional role, food intake is crucial for maintaining glucose levels in individuals with diabetes. Recognizing the specific movements associated with food intake elevates this approach, paving the way for improved diabetes and metabolic disorder treatment methods. The ability to detect these movements and convert them into automated signals for devices such as insulin pumps allows for proactive blood sugar level control. This preventative intervention can significantly enhance diabetes management, leading to better long-term health outcomes and an improved quality of life.

This research advances food consumption detection using deep learning by analyzing the complexities of human behavior to decode the subtle movements that accompany eating. Deciphering food consumption through machine learning represents not just a technological advancement but also a significant shift in healthcare. In this article, I thoroughly examine the rationale, methodology, challenges, and potential impact of using deep learning for food consumption detection. My aim is to emphasize the critical role that technology-driven gesture recognition [R106] [R107] [R108] [R109] can play in optimizing treatment approaches, redefining patient care, and ultimately advancing the field of medicine. The use of Inertial Measurement Unit (IMU) sensors for detecting food consumption is motivated by several key advantages over

other sensor technologies, such as Continuous Glucose Monitors (CGM) and sound sensors, as highlighted in studies like [R110] and [R111]. IMUs are non-invasive and enhance user comfort because they are typically worn externally on devices like smartwatches, unlike CGMs, which require subcutaneous insertion. This makes IMUs more suitable for long-term use and improves user compliance [R110].

Additionally, IMUs are versatile, as they are integrated into widely available consumer devices, reducing the need for specialized equipment. They provide rich, reliable data by capturing detailed movement patterns associated with eating, enabling robust and accurate detection algorithms. In contrast, sound sensors, as explored in [R111], can be inconsistent due to environmental noise and issues with sensor placement [R111].

Figure 3.13 illustrates the gesture detection process. It's important to detail each step of this process. Initially, hand movements (A) generate raw data, which can be measured using accelerometers and magnetometers (B). A smart bracelet or watch (C) is required to extract meaningful information from this raw data. The device collects and transforms the raw data (B) into meaningful values (D). This data is then processed by a deep learning model (E) to determine whether the sequence of movements is associated with a meal. The final output (G) indicates whether the participant is eating (F).

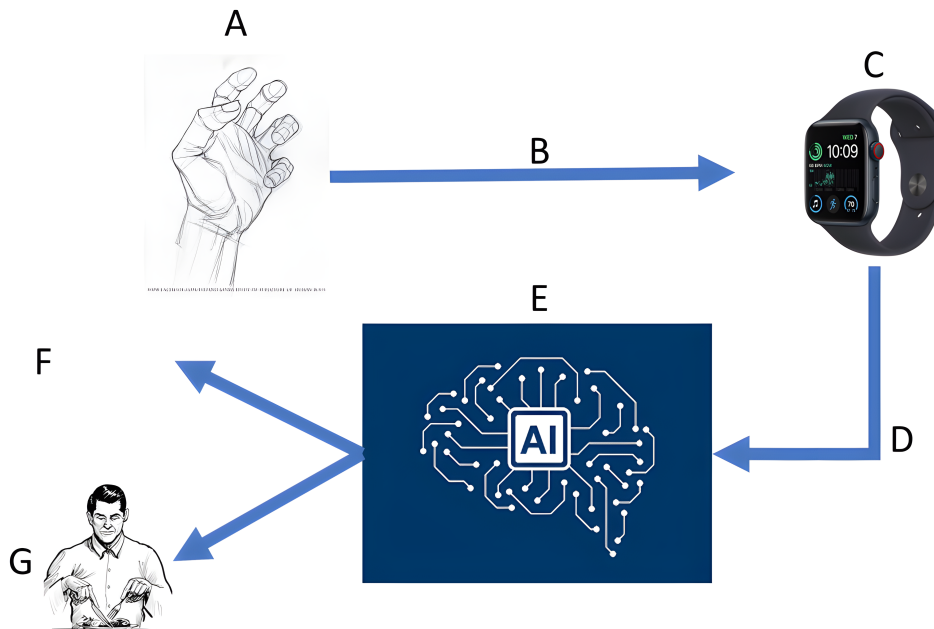


Figure 3.13: Food Consumption Detection system in plan, A: Hand gesture, B: Raw sensor data, C: Smart Watch/Band, D: Collected-Transformed data, E: Deep learning model, F: No Eating, G: Eating

3.4.3 Data collection

Sensor Setup and Data Acquisition for Meal Detection

My research involved testing and utilizing multiple sensor platforms for data acquisition, which was necessary due to several important considerations. Primarily, different devices are equipped with accelerometric sensors that vary in sensitivity, sampling frequency, and measurement range. Additionally, sensor data can vary significantly depending on the orientation and placement of the device on the user's wrist. These factors greatly influence the quality and characteristics of the collected data, which formed the foundation for training artificial intelligence models.

The diversity and richness of the dataset both in temporal resolution and precision were crucial for enabling robust model learning. For example, if meal-related motion was recorded at a low sampling rate, the neural network may not have had sufficient resolution to capture informative patterns. At the same time, it was important to minimize noise and inconsistency in the data.

To begin, I used a sensor unit controlled by an Arduino microcontroller. This setup featured an MPU6050 module, which includes a 3-axis accelerometer and a gyroscope [R112]. The sampling frequency was configured to 75 Hz, based on similar values used in related studies. The sensor data was recorded with precise timestamps and saved to an SD card. The MPU6050 contains a 16-bit analog-to-digital converter, enabling high-resolution measurements. The gyroscope supports a range of 250 to 2000 degrees per second, while the accelerometer measures within a $\pm 2\text{G}$ to $\pm 16\text{G}$ range. Given that eating involves relatively smooth, repetitive motion, these ranges were found to be adequate for the task.

In parallel, I employed the MetaMotionS wristband developed by MbientLab, a device tailored for gesture detection and logging. It incorporates a 6-axis Bosch BMI270 IMU sensor and supports Bluetooth Low Energy (BLE) communication. The device features a 100mAh battery, offering up to 48 hours of continuous operation. The sampling rate of the accelerometer ranges from 0.78 Hz to 1600 Hz, while the gyroscope supports between 25 Hz and 3200 Hz. I interfaced with the device via MbientLab's Python API on an Ubuntu Linux platform and developed custom software for real-time data acquisition and transfer to a laptop.

A critical aspect of the experiment was verifying the sensor's ability to maintain the set sampling rate over long periods. To evaluate this, I compared the expected sample count against the timestamps over extended intervals. The device maintained the configured frequency with an error margin of approximately $\pm 0.6\text{-}0.7\%$, which was acceptable for the purposes of my study.

From empirical observations, I noted that measured values were influenced by individual characteristics such as handedness and eating style. Therefore, metadata was recorded during each session, including whether the subject used their dominant hand, the eating utensil (hand, spoon, fork, knife), and the orientation of the wrist sensor (inner or outer wrist). These variations allowed me to assess the robustness of the trained model to changes in motion direction and context.

In addition, I utilized the Panoramic wristband [R113], a device equipped with sensors similar to those in previous tools but enhanced with a magnetometer for gyroscope calibration. Designed for clinical and research use, this device offers 1 GB of onboard memory for data logging without requiring continuous wireless transmission. Data could be retrieved via BLE using the Android "nRF Connect for Mobile" application. Although the Panoramic device does not support continuous streaming, it transmits data in one-minute intervals suitable for sliding window analysis. The wristband also includes a temperature sensor to detect removal events, automatically pausing data collection and conserving energy.

Due to limited availability of these dedicated sensors, I also explored a smartphone-based solution. I used the "AndroSensor" mobile application to collect accelerometer and gyroscope data from a standard smartphone. The data was stored in CSV format and later processed using custom Python scripts. This allowed me to expand the dataset further by capturing a wide range of real-world motion patterns related to meal consumption.

Each data entry contained acceleration, linear acceleration, and rotational velocity values, along with a timestamp. This allowed for the labeling of meal events based on contextual timing and observation. Data collection was conducted collaboratively, with assistance from several colleagues who contributed by recording their own eating sessions, including times of both meal and non-meal activities.

The final dataset used for training included three acceleration axes and a binary label indicating the presence or absence of a meal. I was able to record 40 meal sessions, resulting in approximately 110,000 data points. Of these, around 92,000 were associated with confirmed

Aspect/Technique	Logistic regression	MLP (30,30)	MLP (30,30,30)	MLP (50,50,50)	MLP (100,100,100)
Accuracy score	0.6715	0.8722	0.8825	0.8870	0.8879
F1-score	0.8018	0.87	0.88	0.88	0.89

Table 3.9: Accuracy results with logistic regression and MLP

Aspect / Technique	RFC (est: 10)	RFC (est: 50)	RFC (est: 100)	RFC (est: 200)	RFC (est: 500)
Accuracy score	0.8706	0.8788	0.8805	0.8807	0.8810
F1-score	0.87	0.88	0.88	0.88	0.88

Table 3.10: Accuracy results with random forest

meal-related movements, while the remaining 18,000 captured hand motions from unrelated daily activities. This carefully constructed and diverse dataset played a pivotal role in training a reliable and generalizable meal detection model.

3.4.4 Machine learning based approach

I conducted tests utilizing various [C4]techniques and methods, including logistic regression, multi-layer perceptron (MLP) networks [R114, R83], and random forest. These methods were applied to datasets comprising the following data:

- ACCELEROMETER X (m / s²)
- ACCELEROMETER Y (m / s²)
- ACCELEROMETER Z (m / s²)
- Label (0 or 1)

For each network, I employed a training-to-test set ratio of 75 (train) / 25 (test). I deemed this ratio appropriate given the ample size of my dataset during the research period, thus favoring a larger number of training samples.

3.4.5 Result with machine learning approach

During the data collection phase, I assembled a dataset comprising over 40 meals with 110,000 records, upon which the aforementioned data manipulation procedures were applied. Out of the 110,000 records, approximately 92,000 records documented eating events, while the remaining 18,000 records captured other similar activities (e.g., toothpicking, smoking). I compared the listed techniques based on various metrics, with the primary considerations being the weighted average F1 score and the Accuracy score. Naturally, MLP and random forest models were experimented with using different parameters to ascertain whether larger networks yielded superior performance. The results of these tests are summarized in Tables 3.9 and 3.10.

The optimal efficiency, reaching 89%, was achieved by a relatively large MLP network. Looking ahead, my aim is to explore techniques capable of consistently detecting meal-related movements even when operating with real-time data.

3.4.6 Clemson dataset

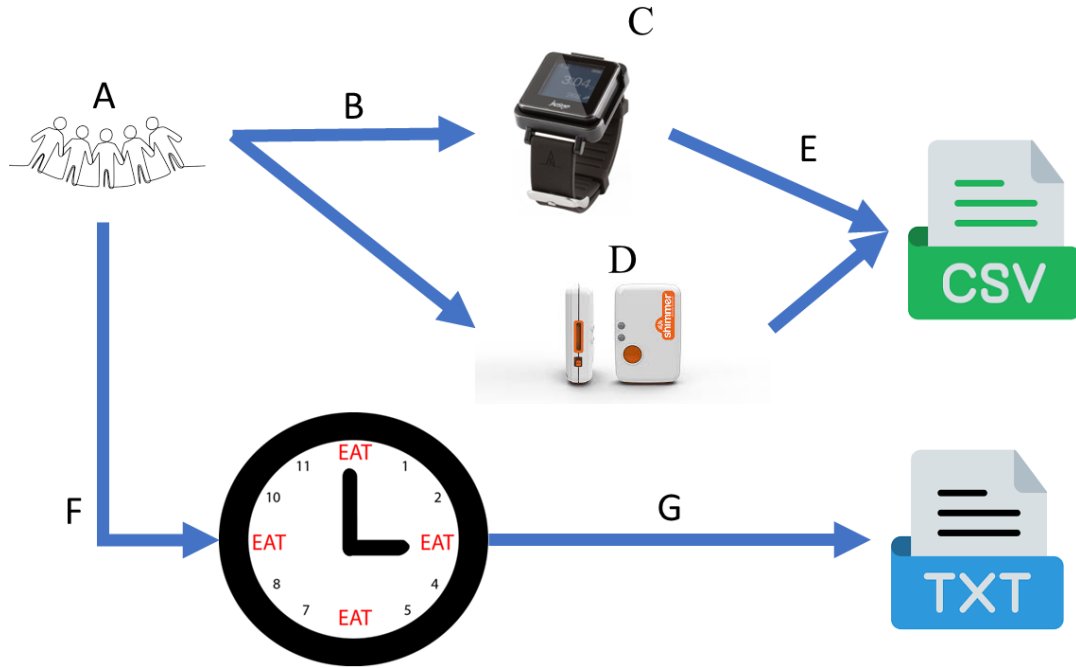


Figure 3.14: Clemson All-day dataset data collection road map A: Patient Population, B: Raw data collection, C: ActiGraph gt9x, D: Shimmer 3, E: Export Data, F: Self reported eating, G: Exported data

The data collection process in the Clemson All-Day dataset, as depicted in Figure 3.14, involved several components. Firstly, a large cohort of patients (A) participated in the study. These patients were equipped with motion sensors, which could be categorized into two main types: ActiGraph gt9x (C) and Shimmer 3 (D). Additionally, other measurement devices were utilized to capture raw data (B) pertaining to hand movements, which were subsequently saved in CSV format (E). These CSV files contained data from accelerometers, magnetometers, and gyroscopes.

On another aspect of data collection, information regarding meals was recorded. Patients themselves documented meal times (F), including details such as the type of meal consumed, its contents, and whether the meal was eaten alone. This meal-related data was stored in a TXT file (G).

The utilization of the Clemson All-Day dataset stands out as a key aspect of this research, offering an extensive, round-the-clock documentation of wrist movement data. Past studies leveraging this dataset have applied sliding window methodologies and convolutional neural networks (CNNs) to derive the continuous probability of food consumption ($P(E)$) from the movement data. Expanding upon this foundation, the daily pattern classifier operates on sequences of $P(E)$, referred to as "daily patterns," facilitating a more comprehensive examination of food consumption trends.

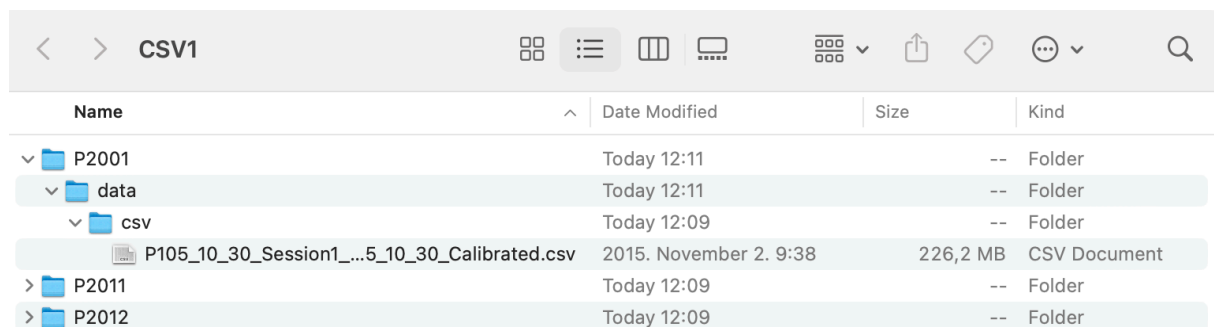
For my research, I employed the Clemson All-Day dataset, comprising wrist movement tracking data from 351 individuals leading their daily lives. Participants utilized wrist movement trackers to delineate meal episodes' onset and conclusion throughout the day. This dataset encompasses raw data sourced from accelerometers, gyroscopes, and magnetometers, alongside processed linear accelerometry data. Offering distinctive insights into daily activities and food consumption behaviors, this dataset enables researchers to delve into movement patterns and their correlation with meals and snacks. This dataset was specifically curated for gesture detec-

tion related to food consumption. Its extensive nature and continuous measurement make it a widely utilized resource in the scientific community. A total of 351 subjects contributed to the data collection, resulting in 354 days' worth of successfully recorded data. Meal reporting was self-initiated by participants through button presses. Various devices were employed for data collection, primarily the Shimmer 3 activity meter, alongside the Actigraph gt9x and iPhones. The dataset boasts significant diversity among subjects, encompassing various ages, races, body weights, and dominant hand preferences. This diversity renders it an invaluable resource for research purposes. Each patient's meal records are stored in separate text files, identified by a unique patient identifier (P####), facilitating precise patient differentiation and analysis.

```
START 2015-10-30 10:02:01
Breakfast 10:48:41 10:58:11 Home NoSeconds Alone Apple_Almonds_Pretzels_and_String_Cheese
NoActivityInfo
Lunch 12:44:00 12:53:31 Restaurant NoSeconds Alone Fish_Broccoli_and_Banana Eating
Dinner 17:09:06 17:39:33 Restaurant NoSeconds CompanyNotEating Sandwich_Salad_Bread_Brownie_Water
Eating
DinnerSnack 22:02:45 22:07:34 Home NoSeconds Alone Wendys_Chilli_and_Fries Eating
END 2015-10-30 23:34:01
```

Figure 3.15: Food consumption txt example

The dataset operators have executed their task commendably, offering two distinct types of data. Firstly, they've provided a downsized dataset synchronized precisely to 15 Hz, which is more manageable in size. Moreover, they've made accessible the raw data captured by the Shimmer instrument. Due to its considerable size, this raw data is partitioned into five segments, constituting a total dataset size of approximately 20 gigabytes. Each segment, comprising about 4 gigabytes, encompasses patient data tagged with unique patient IDs.



Name	Date Modified	Size	Kind
✓ P2001	Today 12:11	--	Folder
✓ data	Today 12:11	--	Folder
✓ csv	Today 12:09	--	Folder
P105_10_30_Session1_...5_10_30_Calibrated.csv	2015. November 2. 9:38	226,2 MB	CSV Document
> P2011	Today 12:09	--	Folder
> P2012	Today 12:09	--	Folder

Figure 3.16: Directory structure

As depicted in the image, within this folder structure, you'll encounter patient folders containing data. Navigating to the CSV directory, you'll discover CSV files recorded by individual patients. These files house raw data obtained from the measurement tool. Notably, the delimiter within the CSV file is not a comma but a tab. The dataset comprises 17 columns, each column featuring the measurement name along with the corresponding unit in the subsequent row. The columns within the complete CSV file are listed as follows:

- P105_10_30_Timestamp_CAL msec
- P105_10_30_RealTime_CAL msec
- P105_10_30_GSR_CAL kOhms
- P105_10_30_Quat_MPL_6DOF_W_CAL no units
- P105_10_30_Quat_MPL_6DOF_X_CAL no units

P105_10_30_Timestamp_CAL	P105_10_30_RealTime_CAL	P105_10_30_GSR_CAL	P105_10_30_Quat_MPL_6DOF_W_CAL
msecs	msecs	kOhms	no units
73853855.8654785	1446213721684.85	-40.0136046255727	0.988325893878937
73853922.5158691	1446213721751.5	-40.0136046255727	0.988035380840302
73853989.1662598	1446213721818.15	-40.0136046255727	0.987957119941711
73854055.8166504	1446213721884.8	-40.0136046255727	0.988576531410217
73854122.467041	1446213721951.45	-40.0136046255727	0.988661766052246
73854189.1174316	1446213722018.1	-40.0136046255727	0.989851534366608
73854255.7678223	1446213722084.75	-40.0136046255727	0.992486119270325
73854322.4182129	1446213722151.4	-40.0136046255727	0.992896437644959
73854389.0686035	1446213722218.05	-40.0136046255727	0.992318570613861
73854455.7189941	1446213722284.7	-40.0136046255727	0.992841184139252
73854522.3693848	1446213722351.35	-40.0136046255727	0.992006182670593
73854589.0197754	1446213722418.0	-40.0136046255727	0.991322696208954

Figure 3.17: CSV file example

- P105_10_30_Quat_MPL_6DOF_Y_CAL no units
- P105_10_30_Quat_MPL_6DOF_Z_CAL no units
- P105_10_30_Gyro_MPU_MPL_X_CAL deg/sec
- P105_10_30_Gyro_MPU_MPL_Y_CAL deg/sec
- P105_10_30_Gyro_MPU_MPL_Z_CAL deg/sec
- P105_10_30_Accel_MPU_MPL_X_CAL m/(sec²)
- P105_10_30_Accel_MPU_MPL_Y_CAL m/(sec²)
- P105_10_30_Accel_MPU_MPL_Z_CAL m/(sec²)
- P105_10_30_Mag_MPU_MPL_X_CAL uTesla
- P105_10_30_Mag_MPU_MPL_Y_CAL uTesla
- P105_10_30_Mag_MPU_MPL_Z_CAL uTesla
- P105_10_30_Date dd_MMM_yyyy_HH:MM:SS.FFF

As evident, the columns represent session names, complicating the data conversion process necessitating truncation. The subsequent image provides a brief glimpse of the raw data's appearance. Due to its extensive nature, displaying the entire dataset would be impractical, hence, I've omitted the initial columns for brevity.

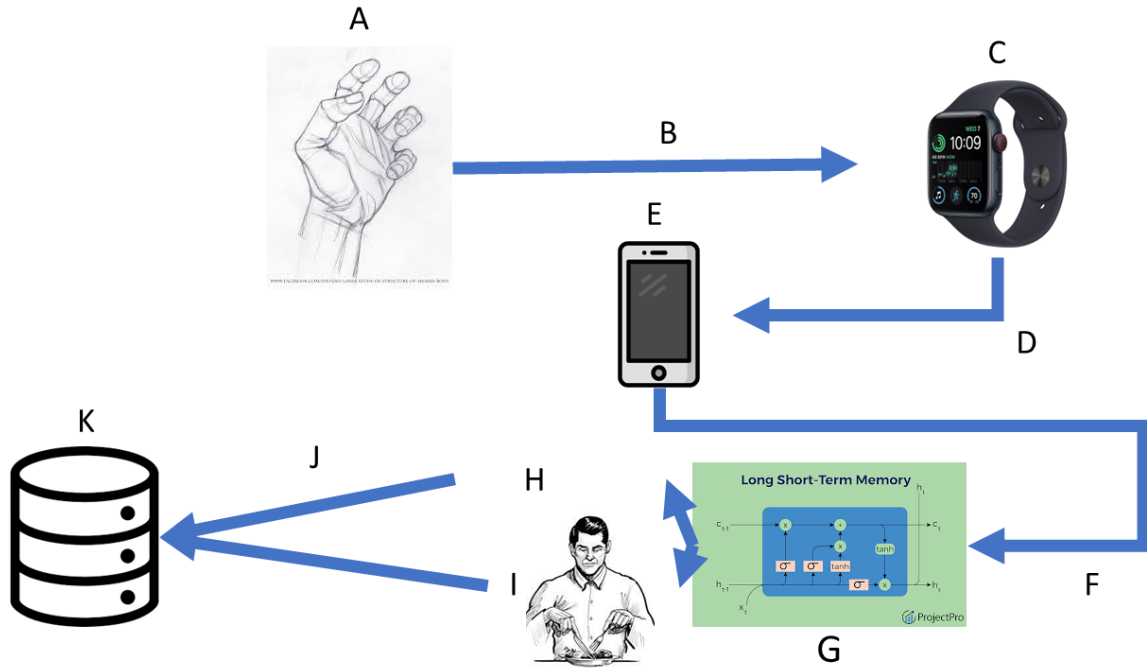


Figure 3.18: My proposed method in real life use, A: Hand gesture, B: Raw sensor data, C: Smart Watch/Band, D: Collected data, E: Smart Phone, F: Transformed data, G: LSTM modell, H: No Eating, I: Eating, J: Export, K: Database

In Figure 3.18, you can observe the practical application of the developed model. As depicted in Figure 3.13, hand movements (A) serve as the initial input, generating raw data (B) for recording. Optimal recording is achieved through a smartwatch or smart bracelet (C), with data transmission facilitated to a smartphone (E) via Bluetooth technology (D). The smartphone performs data conversion into the requisite format (F), creating one-minute data stacks, as outlined. Within this dataset, elements are sequenced every second. The transformed feature vector is then inputted into my LSTM model (G) for decision-making. Remarkably, this LSTM model can be integrated into a smartphone application for on-the-go usage. Subsequently, the model output discerns whether the motion sequence corresponds to a meal (I) or not (H). This data is exportable (J) to a database (K), which could also manifest as a logging application. The data was stored in CSV format, with each CSV file containing a session representing either food consumption or another activity. To extract the data, 12 data points were selected from the raw data. Initially, I examined the frequency to assess any deviations from the 15 Hz frequency, which the meter could measure. Upon testing, I found no deviations, indicating that all data adhered to the correct frequency without errors. The presence of real-time data in the second column facilitated the calculation of intervals between two time points. Subsequently, since the sampling rate was 15 Hz, data was extracted at this frequency and averaged to obtain raw data per second. Working with data in seconds simplified labeling tasks. Additionally, uncertainty regarding the model's prediction speed influenced my decision to extract data at this frequency. The sliding window approach employed was not one-by-one; instead, a new second was considered every 15 seconds.

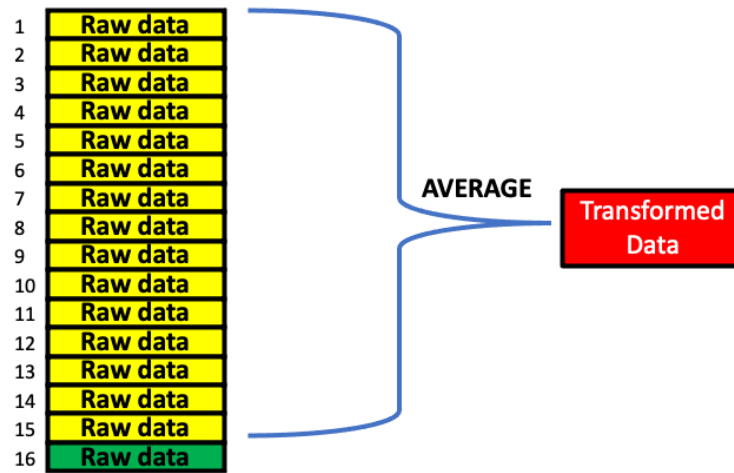


Figure 3.19: Data aggregation

Subsequently, I extracted meal data reported by the patient. This involved parsing through the file line by line to identify meal entries. Upon encountering a meal entry, I recorded the start and end points of the meal. These timestamps were then converted from string format to timestamp format, facilitating data processing in seconds. This streamlined the subsequent data processing and labeling tasks. Data falling outside the meal's start and end times was discarded, while the remaining data was initially labeled with a value of 0. Next, I iterated through the list of meals, and for each meal item, I traversed the extracted second data. If a transformed timestamp of a data element fell within the meal's start and end time, it was labeled with 1.

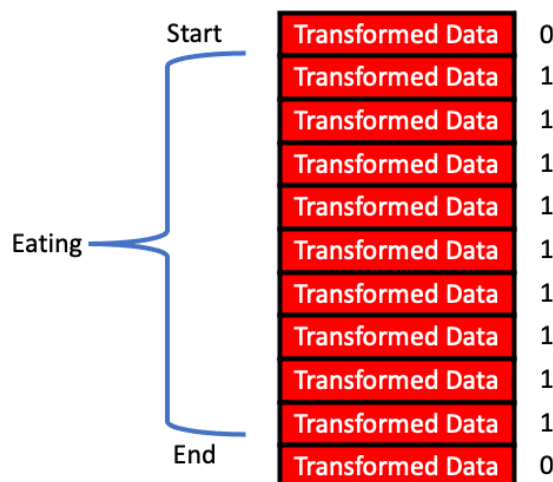


Figure 3.20: Labelling

Subsequently, I filtered out any raw data components deemed unsuitable for training due to their potential patient-specific influence on the outcome. Specifically, I excluded two timestamp values, as not all participants measured on the same day or initiated measurements simultaneously. Additionally, the date attribute was omitted due to its potential patient-specific nature. Furthermore, the GSR attribute, measured in kOhm, was discarded as it pertained to device specifics. Finally, I dropped the Quat W CAL attribute, as the Quat values were count-based, aligning with my decision to exclusively work with X, Y, and Z axis data. Consequently, I retained 3 Quat values, 3 gyroscope values, 3 accelerometer values, and 3 magnitude values, alongside a label denoting meal presence or absence, totaling 13 fields (12 attributes and one label). This curated dataset was saved in CSV format files, with separate files created for each of the 354 days, forming the basis for training.

	Quat X	Quat Y	Quat Z	Gyro X	Gyro Y	Gyro Z	Accel X	Accel Y	Accel Z	Mag X	Mag Y	Mag Z	label
0	-0.037029	-0.143409	-0.020278	6.601624	13.617981	-12.149480	0.207723	-0.039358	0.950165	7.588842	14.853607	-27.370056	0.0
1	0.010175	-0.143176	-0.014098	1.735087	-2.499369	5.086848	0.135758	0.056419	0.963682	10.588720	12.593699	-27.280060	0.0
2	0.009881	-0.225304	-0.002689	-2.187439	-24.903341	-1.773504	0.211206	0.033250	0.924359	6.308894	12.753693	-28.580007	0.0
3	0.004284	-0.344391	-0.063012	-21.340271	-61.011739	-22.212143	0.467342	0.104677	0.707595	-5.290634	12.593699	-24.800161	0.0
4	-0.436980	-0.652124	-0.098807	19.736389	57.587708	37.834244	0.715202	-0.438015	-0.332599	-17.810125	30.892955	14.498240	0.0
...
48708	0.014149	0.142930	-0.797766	4.761316	-5.624698	1.633152	0.163257	-0.222194	0.928379	19.625285	7.451462	-38.239543	0.0
48709	0.002756	-0.034559	-0.814554	14.172946	8.112196	5.201722	-0.024618	0.082233	0.959377	25.465047	-3.088109	-32.879761	0.0
48710	-0.002008	-0.053046	-0.842257	-2.339117	0.409623	2.202812	-0.031234	0.114524	0.969419	25.005066	-5.748001	-31.859802	0.0
48711	0.022725	-0.058323	-0.841808	-0.508045	3.156059	-1.353423	-0.093384	0.091038	0.969826	25.805033	-5.288020	-30.899841	0.0
48712	0.038247	-0.065875	-0.835379	1.061497	2.594967	-2.944683	-0.125305	0.093506	0.952087	27.764954	-5.583008	-30.059875	0.0

Figure 3.21: Transformed data for training

Figures 3.22, 3.23, 3.24, and 3.25 display sample data from a single patient, illustrating the four types of sensor readings. The figures include data from the magnetometer, accelerometer, and gyroscope, represented along their three axes, while the quaternion includes three axes and the total calculated displacement. The axis values for each sensor type are plotted individually, providing clearer insights into the behavior of each sensor during carbohydrate intake. The meal periods are indicated in pale red on the graphs.

For the accelerometer, there is little difference in sensor readings between meal and non-meal periods, although the sensor values are notably higher when meals are present. In contrast, the gyroscope shows a significant change, with all axes displaying higher values during meals. Similarly, the magnetometer readings show an opposite trend, with minimal variation during meals. Finally, the quaternion data reveals large fluctuations around mealtimes, with the frequency of these changes noticeably higher compared to non-meal periods.

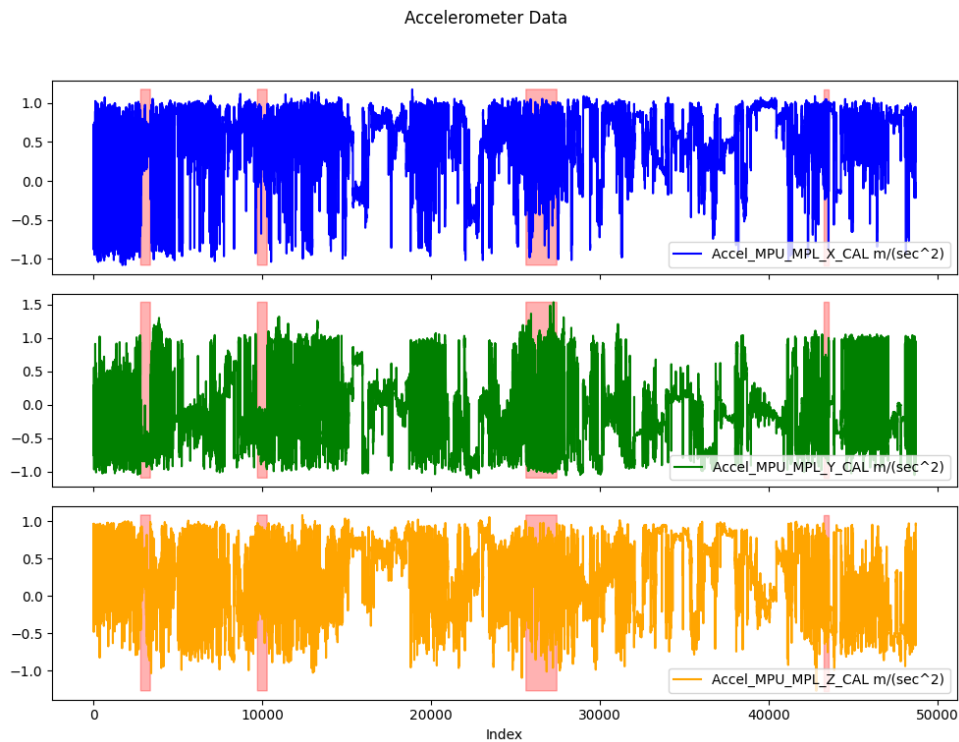


Figure 3.22: Sample patient accelerometer data after transformation

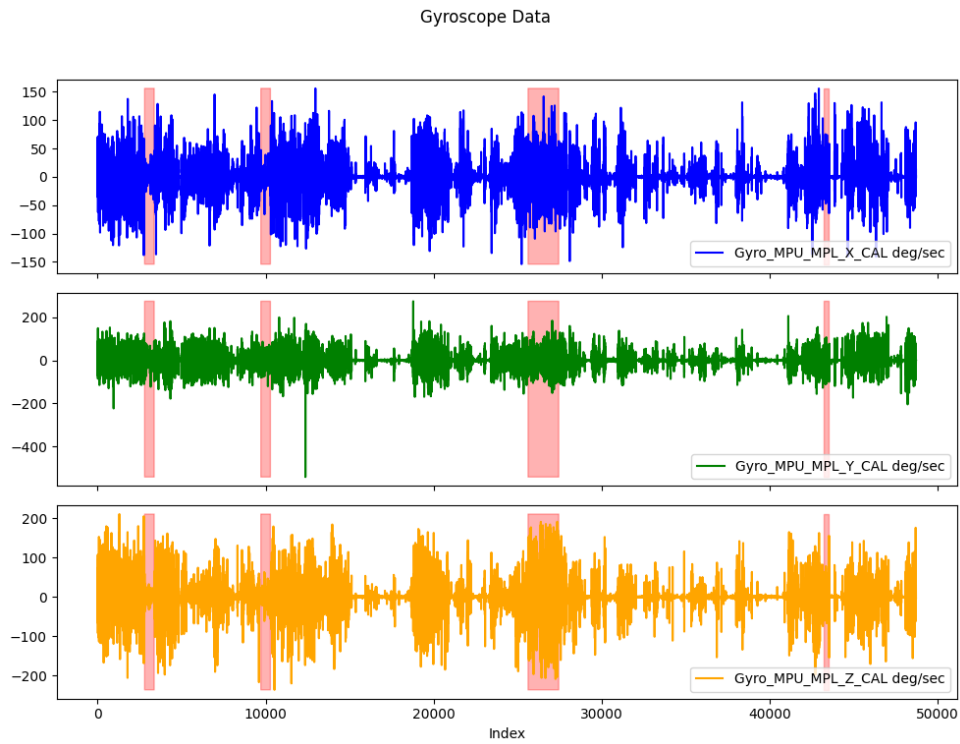


Figure 3.23: Sample patient gyroscope data after transformation

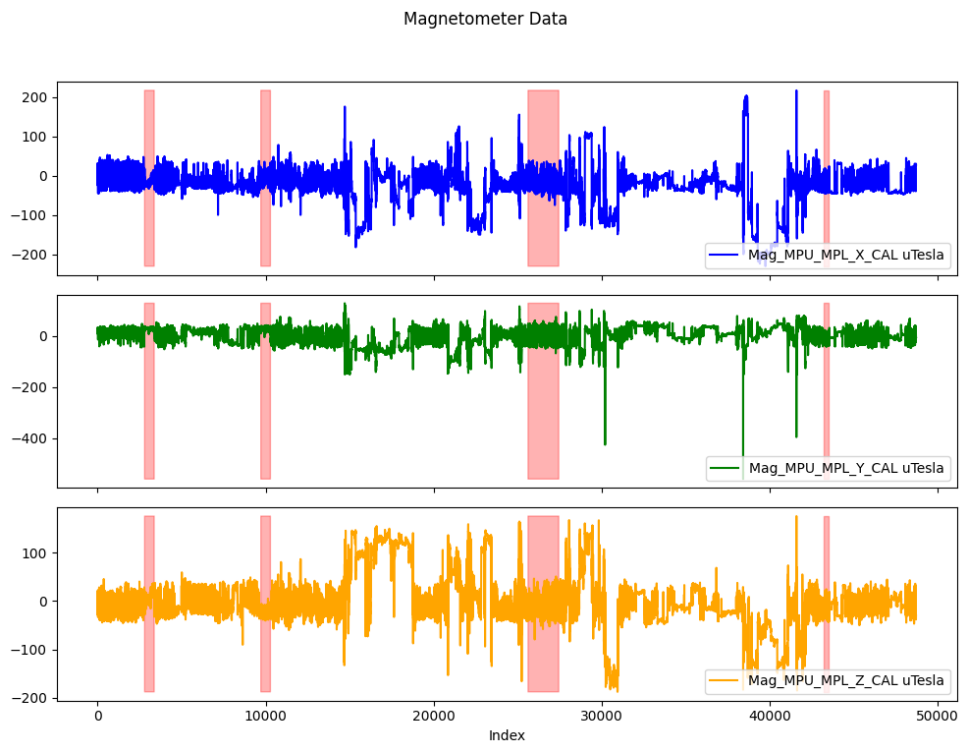


Figure 3.24: Sample patient magnetometer data after transformation

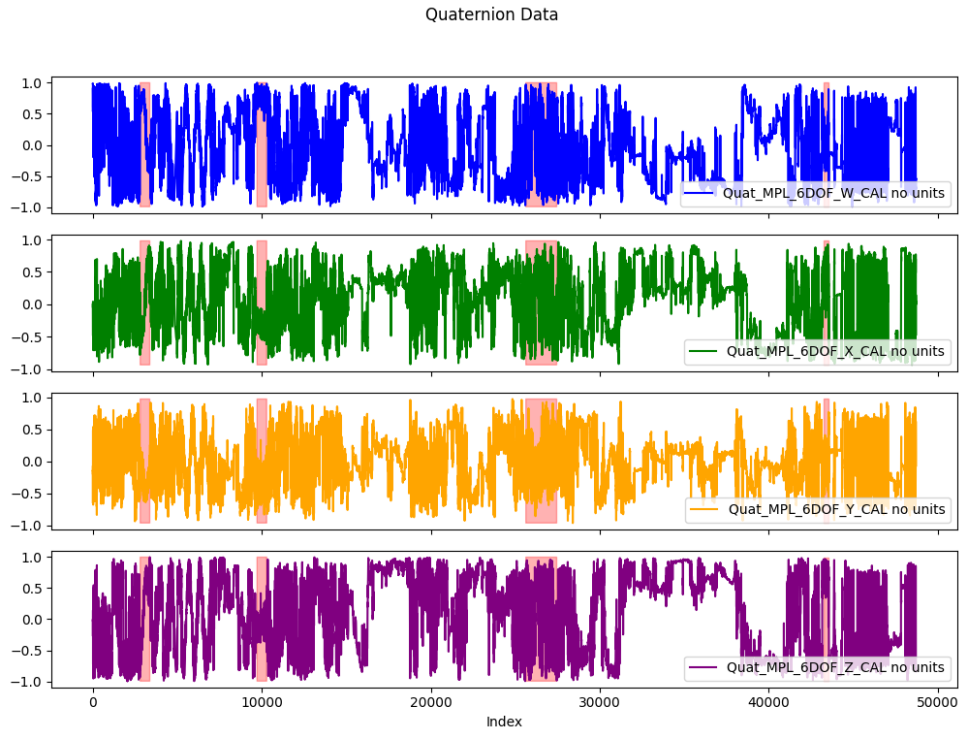


Figure 3.25: Sample patient quaternion data after transformation

3.4.7 LSTM based method

In my previous study, I exclusively utilized simple machine learning algorithms [R115]. However, in my current investigation [J3], I opted for a deep neural network approach. Recurrent Neural Networks (RNNs) inherently possess memory, facilitating information retention. Nonetheless, traditional RNN architectures struggle with effectively recalling long sequences of data. To address this limitation, Long Short-Term Memory (LSTM) networks were introduced [R116]. Accordingly, my model incorporates LSTM layers to enhance memory retention capabilities.

The input to my model comprises 60 vectors, each containing 12 elements, aligning with the one-minute data granularity. I employed two LSTM layers, each comprising 256 neurons, with the `return_sequences` attribute set to true [R117]. Consequently, the LSTM layers produce an output of 60 vectors, each containing 256 elements. To prevent overfitting, a dropout layer with a dropout rate of 0.2 was incorporated [R118].

Subsequently, another LSTM layer with identical configuration followed, alongside a dropout layer with the same dropout rate. A `GlobalAveragePooling1D` layer was then applied to compute the average of the 60 vectors, yielding a vector with 256 elements [R119]. The subsequent layers are responsible for classification.

Preceding the final classification layer are two blocks, each comprising a dense layer with 64 neurons and a Rectified Linear Unit (ReLU) activation function [ReLU]. Following each dense layer is a dropout layer with the same dropout rate mentioned earlier [R118]. The classification layer itself consists of two neurons with a softmax activation function, reflecting the binary classification nature of the task.

Sparse categorical cross-entropy served as the cost function [R120], and the Adam optimizer was utilized for optimization [R121]. The decision to employ only two neurons in the classification layer aligns with the binary classification task at hand.

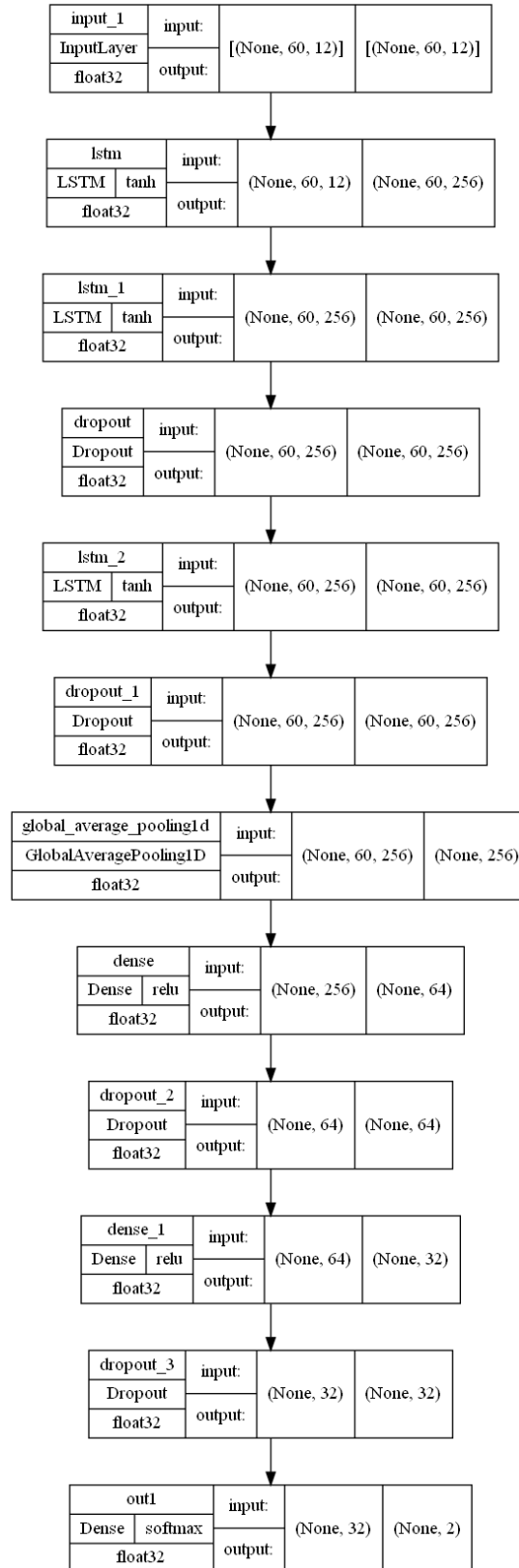


Figure 3.26: LSTM model

3.4.8 Training testing

During training and testing, data were grouped by one minute. In contrast to data extraction, I have now moved my sliding window to every second rather than every minute. That is, I

extracted the first sixty values from the csv file and then I latched the sub-file and extracted 60 values again. I did this until I got to the end of the csv wall for that day. The label for the decision was the value at the 60th element. Since I only wanted to tell for the current time point whether there was an arrival or not. I did not want to make a decision for the other seconds before that. Following the rule for time series data that I do not randomly extract data from a given data set. But by observing the events that occurred in the time series. That is, the first $x\%$ of the data is training data, while the remaining $y\%$ is testing data. So I did a resolution of 80 and 20%. The first 80% of each day was the training sample, while the remaining 20% went to the test data set. The training was done over 100 epochs. During training, the model that achieved the lowest cost function value was eliminated. Then at testing this model was reloaded and performance analysis was performed on it.

3.4.9 Validation

Validation of the results was conducted for one patient at a time, following the established protocols for time series data analysis. Data samples were not randomly selected for training and testing datasets. Instead, the first 80% of the data for each patient constituted the training dataset, while the remaining 20% comprised the test dataset. Metrics were then calculated for each patient based on this test dataset. Subsequently, the results obtained from these metrics were aggregated and presented in the results section. This validation method was employed to ascertain whether a day's sample provided sufficient information to learn a patient's patterns effectively.

3.4.10 Results with LSTM model

Let's delve into the findings derived from the analysis of the tests. Firstly, let's refer to Table 3.11, which provides a comprehensive summary of precision, recall, and F1 score values spanning 354 days. The table not only presents the median and mean of these metrics but also includes the first and upper quartile values, along with the maximum and minimum values. Additionally, the standard deviation from the mean is depicted.

Commencing with the minimum values, it's noteworthy that despite being the lowest, the Recall column still reaches the 80% range. Similarly, the Precision column achieves a minimum of 86%. Examining the F1 score, which amalgamates Precision and Recall, yields an impressive 88%. This metric serves as a robust indicator of model performance. The F1 score's minimum might surpass the minimum values of the other metrics because it's a composite metric, thus even the poorest-performing model can predict with an 88% accuracy.

Moving on to the bottom quartile, which encapsulates the poorest performing 25% of the dataset, both Precision and Recall columns display a strong 99%. Similarly, the F1 score metric reflects a commendable 98%. This suggests that in 75% of the tests, the F1 score exceeds 98%. However, it's crucial to acknowledge the presence of outliers.

Considering the median and mean scores, there's a marginal 1% discrepancy between the median and average in the F1 score and Recall column. This indicates the presence of outliers among the test scores. Nonetheless, on average, the F1 score can reliably achieve approximately 98% to 99%. Utilizing the median, it can be inferred that half of the test scores exceed 99%. The minimal standard deviation implies that the majority of tests cluster closely around the mean.

At the upper quartile or maximum value, I observe a model achieving 100%. This may occur when there are test cases with minimal meal occurrences, leading to no meals being included in the test set during the split.

	Precision	Recall	Fscore
mean	0.990405	0.988249	0.989036
std	0.021567	0.027148	0.018902
min	0.864838	0.816071	0.888330
25%	0.993431	0.991326	0.988496
50%	0.998181	0.998014	0.997057
75%	1.000000	1.000000	0.999205
max	1.000000	1.000000	1.000000

Table 3.11: Description of metrics

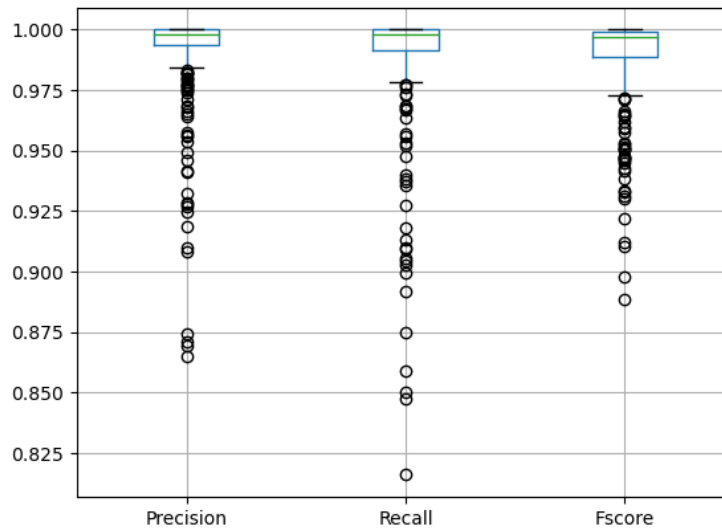


Figure 3.27: Boxplot for metric

Now let's analyze Figure 3.27, which employs boxplot representations to visualize the metric values. The observations I've made thus far are corroborated here. Notably, a considerable number of outliers emerge during the tests, yet none fall below 80%. Specifically, in the case of the F1 score, only two instances fail to reach a 90% performance, indicating that merely two out of 354 items exhibit lower scores.

Examining the body of the boxplot reinforces my initial hypothesis. The test results predominantly cluster around the 98-99 range, aligning with my expectations from the computation. It's also noteworthy that in the precision metric, only three values fall below 90%. This implies that the models generate few false positives, which is advantageous. However, the situation is less favorable for Recall, where this number is double that of Precision. Unfortunately, the models predict more false negatives in this scenario.

Nevertheless, when considering the aggregate boxplot values for all three metrics, it's evident that 11 cases fail to reach 90% performance on any metric. Additionally, a greater number of outliers above 90% performance are observed. Despite this, considering that I haven't achieved 90% performance thus far, the absence of outliers represents a notable improvement.

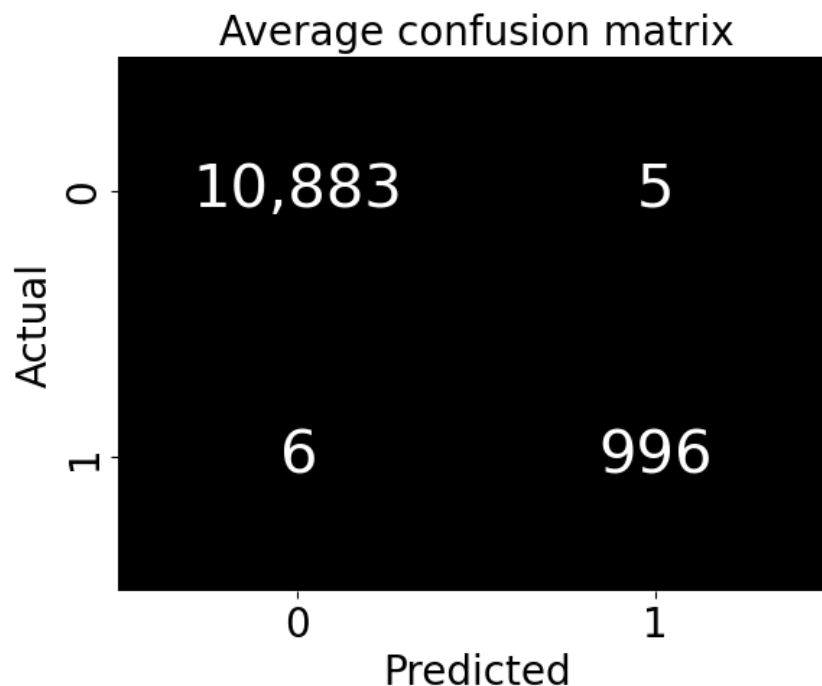


Figure 3.28: Confusion matrix

Finally, let's scrutinize the confusion matrix averaged from the results depicted in Figure 3.28. The confusion matrix serves as a revealing representation of my work with non-smoothed datasets. Given that the confusion matrix provides mean values, any statements made should pertain to the average.

Returning to the non-smoothed dataset, on average, the number of positive samples accounts for 10% of the number of negative samples in the test datasets. However, it's essential to focus on the number of false positives and negatives, which are more relevant to my analysis.

Let's begin with false positives, which occur when the model predicts a meal when there isn't one. This may occur at the end of a meal due to the absence of discernible changes in the data. In this scenario, the average value is 5. This indicates that, on average, the model continues to predict food consumption 5 seconds after a meal has concluded. This value is relatively low, suggesting that at a minute resolution, errors may be minimized.

Conversely, consider the situation where the model fails to predict a meal when one is actually occurring. This typically occurs at the start of a meal, as there may not yet be sufficient changes in the data. In this case, the average value is 6. Hence, on average, it takes 6 seconds from the start of a meal before the model detects that a meal is being consumed.

The average values of 5 and 6 imply that the longer it takes for the model to detect meal consumption, the longer it will take for the model to accurately predict meal consumption. This phenomenon stems from the internal dynamics of the LSTM cell, where changes may not be as pronounced. This issue could potentially be addressed by implementing a model that considers a longer time span and predicts both the beginning and end of a meal.

3.4.11 Conclusion

In summary, the tests indicate that I achieved better results on a more complex dataset compared to the dataset I collected ourselves. Recurrent networks proved particularly effective in solving the problem. However, some outliers remain in the test results. Despite this, the model I developed consistently achieves over 90% accuracy on the dataset, with an average prediction delay of 5.5 seconds. This delay means the model predicts meal events slightly late and extends

the period it identifies as meal time.

It's important to note that while some participants in the dataset have several days' worth of data, most have data for only one day. This raises the question of how the model would perform if multi-day data were required. A potential solution would involve collecting multi-day data, including nighttime periods, and testing the model on that dataset. Furthermore, the 5.5-second latency I observed might be influenced by the one-minute resolution, which could be too broad for the LSTM model to capture changes in the data effectively. Reducing the time window might improve the model's responsiveness.

Overall, I developed a model that demonstrates outstanding performance on this dataset, consistently achieving an accuracy between 98-99%. Future research could enhance this model further by incorporating alternative datasets and applying advanced cross-validation techniques. Collecting a new dataset will be crucial in advancing this work. Additionally, incorporating transformer networks and using shorter time windows could enhance future models. We also plan to experiment with non-personalized models that can be applied in real-world scenarios.

One key limitation of current artificial pancreas systems is the delayed response of insulin, which needs to be administered 15-20 minutes before meals to effectively control postprandial glucose levels. This timing delay reduces the real-time effectiveness of closed-loop systems during meals, underscoring the need for system enhancements. Furthermore, automatic meal detection is essential to reduce reliance on patient input, especially for elderly users, where forgetting meal times could hinder accurate logging.

3.5 Insulin control with reinforcement-based neural network

3.5.1 Goal

The main objective of my research was to investigate the feasibility of tuning the control of the AP (artificial pancreas) device with a reinforcement-based neural network using devices connected to the artificial pancreas. There are already artificial pancreas systems in the world that consist of three main components. The first component is the continuous glucose monitoring sensor (CGMS), which allows the measurement of the patient's blood glucose levels at 5-minute intervals. The second main component is the insulin pump, which can deliver insulin to the patient. The third component is an advanced control algorithm. In this thesis I present a modern adaptive example of this algorithm. This solution has a number of advantages, allowing for a much more personalised therapy design and eliminating the need for a longer therapy adjustment period and the need for experimentation. The patient can wear the device and in the longer term learn the ideal dosage on his/her own - and initially it will deliver the dosage in a reliable way (acceptable to the patient, but not personalised). In the initial phase of my research, I conducted basic tests and learned about reinforcement learning (RL). I tested a selected mathematical model using a virtual patient simulator. The simulator provided the environment for reinforcement learning, while the model or neural network represented the agent. The selected mathematical model was a simple-structure IVP model calibrated to 10 real patients and used in an environment of 17 virtual patients. At each step I ran the mathematical model, which returned a measured blood glucose value. Based on this, my neural network model calculated how much insulin to administer in a given iteration.

3.5.2 Introduction

Diabetes mellitus (DM) is a chronic metabolic disease that remains incurable, arising from either a complete lack of insulin in Type 1 Diabetes Mellitus (T1DM) or a partial deficiency and/or inadequate effect of insulin in Type 2 Diabetes Mellitus (T2DM). The precise pathophysiology is not fully understood; however, T1DM is thought to result from a series of autoimmune reactions that destroy the insulin-producing cells in the pancreatic islets of Langerhans [R122]. This

manuscript focuses on managing blood sugar levels in T1DM using reinforcement learning-based artificial intelligence techniques for external insulin administration. T1DM typically develops rapidly, primarily affecting children and adolescents who may have a genetic predisposition. However, lifestyle factors and external circumstances can also accelerate the onset of the condition [R123, R124]. The primary molecular function of insulin is to facilitate the entry of glucose into insulin-sensitive tissues, such as muscle and adipose tissue, and to suppress excessive glucose production (gluconeogenesis and glycogenolysis) primarily in the liver and kidneys [R32]. Patients with T1DM require lifelong external insulin to maintain their blood glucose levels [R125]. Without this external insulin, they cannot survive due to their body's inability to regulate energy metabolism [R126]. Robust, high-quality, and personalized blood glucose management is crucial for patients with T1DM to achieve normal glycemia and reduce the risk of complications. Affected individuals often develop long-term complications, such as damage to the retina, microvascular system, nerves, and kidneys [R127, R128, R129].

In recent years, semi-automated insulin administration for T1DM has been shown to maintain satisfactory glycemia over time and reduce the risk of serious complications [R130, R131]. However, there is a significant need for further development of control algorithms for insulin administration. These solutions must be robust initially and possess self-tuning capabilities to adapt to the patient's needs over time, providing personalized treatment and insulin delivery [R38]. Semi-automatic glycemic control typically follows the artificial pancreas (AP) concept, which consists of three key components: a Continuous Glycemia Monitoring System (CGMS) for measuring blood sugar levels, an insulin pump to deliver insulin, and an advanced control algorithm [R38, R39, R132, R133].

An AP system is designed to monitor blood glucose levels and automatically administer insulin to manage diabetes within certain parameters. It comprises a continuous glucose monitor, an insulin pump, and a control algorithm. Studies, such as [R134], demonstrate its effectiveness in improving glucose control and reducing episodes of hypoglycemia. These systems, approved by the FDA, provide a more automated approach to diabetes management.

Control algorithms within artificial pancreas systems are crucial for determining insulin doses based on real-time glucose data [R135, R136, R137, R138]. These algorithms continuously analyze glucose sensor readings and utilize mathematical models to predict future glucose levels. The main objective is to keep blood sugar within a target range while minimizing the risks of hypoglycemia (low blood sugar) and hyperglycemia (high blood sugar). Over the past two decades of artificial pancreas (AP) development, a variety of control algorithms have been created and studied, including proportional-integral-derivative (PID) controllers, model predictive controllers (MPC), and fuzzy logic controllers. Each algorithm has its advantages and limitations, and researchers are continuously working to refine and optimize them for improved performance and safety [R139, R140, R141, R142, R143, R144].

The selection of a control algorithm depends on factors such as an individual's insulin sensitivity, meal intake, physical activity, and overnight glucose patterns. Additionally, advancements in machine learning and artificial intelligence have led to the creation of more adaptive and personalized algorithms that can adjust insulin dosing based on individual variability and changing circumstances [R145, R146]. Current blood glucose management methods for type 1 diabetes involve regular self-monitoring of blood glucose levels and insulin administration. These methods, however, demand considerable commitment from patients. Consequently, there is growing interest in applying machine learning techniques, especially RL, to improve diabetes management [R147, R148, R149]. RL is an advanced machine learning method in which an agent-like algorithm (often a neural network-based model) learns the optimal policy for taking actions by receiving feedback in the form of rewards or penalties. Its effectiveness has been demonstrated in various fields, including game playing [R150], robotics [R151], and healthcare [R152]. In blood glucose management, RL can be used to develop a personalized strategy for adjusting insulin dosages by analyzing current and past blood glucose levels, insulin doses, and

other relevant factors [R153].

3.5.3 Virtual Patient model

The foundation for the controlled environment was established using the Identifiable Virtual Patient model [R154].

These parameter sets can be examined in the Table 3.12.

BW	GEZI	EGP	CI	SI	tau1	tau2	p2	VG
89	3.87E-08	1.4	2010	4.93E-04	49	47	1.06E-02	253
89	2.20E-03	1.33	2010	8.11E-04	49	47	1.06E-02	253
63	4.38E-03	0.6	1281	9.64E-05	41	10	1.16E-02	261
65	3.50E-03	0.856	909	1.70E-04	71	70	2.33E-02	199
65	1.64E-03	1.07	909	4.63E-04	71	70	2.33E-02	199
116	7.58E-08	2.59	1813	3.77E-04	91	70	8.14E-03	337
116	1.64E-05	0.98	1813	3.77E-04	91	70	8.14E-03	337
64	4.33E-03	0.6	1535	2.05E-04	46	46	9.63E-03	188
51	1.01E-03	0.603	588	4.12E-04	68	30	9.15E-03	104
77	2.30E-03	1.11	1806	8.16E-04	60	60	1.01E-02	263
65	1.00E-08	1.3	540	3.68E-04	95	37	1.03E-02	137
100	6.39E-03	1.27	875	2.56E-04	131	21	1.03E-02	193
64	1.04E-03	0.611	1309	6.03E-04	53	53	1.02E-02	204
51	3.79E-03	0.603	588	9.48E-04	68	30	9.15E-03	104
65	1.00E-08	0.601	540	5.40E-04	95	37	1.03E-02	137
100	6.39E-03	3.45	875	6.89E-04	131	21	1.03E-02	193
64	1.04E-03	0.611	1309	1.73E-03	53	53	1.02E-02	204

Table 3.12: Patient parameter sets used in the Identifiable Virtual Patient (IVP) model. Rows with identical values for VG and BW represent data from the same patient. The parameters include BW (Body Weight), GEZI (Gastric Emptying and Insulin Sensitivity), EGP (Endogenous Glucose Production), CI (Clearance Index), SI (Sensitivity Index), tau1 (Delay Parameter 1), tau2 (Delay Parameter 2), p2 (Parameter 2), and VG (Volume of Glucose). In the case where the parameters (BW) and (VG) are the same, I are talking about the same patient. But with night and day parameter sets.

In my simulation environment, this model was expanded to include a representation of sensor noise. While the inclusion of GEZI in the model may be subject to debate, it serves as a commonly employed, straightforward model of glucose-insulin metabolism, and I opted for its use for illustrative purposes. My methodology involved the integration of a cohort of 10 uniquely characterized patients, supplemented by an additional set of parameters tailored for nighttime periods. Consequently, a total of 17 distinct virtual patient profiles were generated. The dynamics of a patient are described by the following equations:

$$\dot{G}(t) = -(GEZI + I_{EFF}(t)) \cdot G(t) + EGP + R_A(t) \quad (3.18)$$

$$\dot{I}_{EFF}(t) = -p_2 \cdot I_{EFF}(t) + p_2 \cdot S_I \cdot I_P(t) \quad (3.19)$$

$$\dot{I}_P(t) = -\frac{1}{\tau_2} I_P(t) + \frac{1}{\tau_2} I_{SC}(t) \quad (3.20)$$

$$\dot{I}_{SC}(t) = -\frac{1}{\tau_1} I_{SC}(t) + \frac{1}{\tau_1 C_I} u(t) \quad (3.21)$$

$$R_A(t) = \sum_i^m \frac{d_i}{V_G \cdot \tau_{D_i}^2} t_i \cdot e^{-\frac{t_i}{\tau_{D_i}}} \quad (3.22)$$

Blood glucose concentration is denoted as $G(t)$, measured in milligrams per deciliter (mg/dL), while the effectiveness of insulin is represented by $I_{EFF}(t)$, measured in units of inverse time (min^{-1}). Subcutaneous and plasma insulin concentrations are symbolized as $I_{SC}(t)$ and $I_P(t)$ respectively, both expressed in microunits per milliliter ($\mu U/mL$). The agent indirectly influences blood glucose levels through insulin infusion, while disturbances in the form of carbohydrate intakes d_i in grams are encapsulated by the parameter R_A .

The timing parameters τ_1 and τ_2 are measured in minutes, while the rate constant p_2 is expressed in units of inverse time (min^{-1}), defining the absorption kinetics of insulin. Insulin clearance is denoted as C_I , measured in milliliters per minute (mL/min), and insulin sensitivity is represented by S_I , expressed in milliliters per microunit per minute ($mL/\mu U/min$).

Endogenous glucose production, symbolized as EGP , quantifies liver glucose output and is measured in milligrams per deciliter per minute ($mg/dL/min$). Glucose effectiveness at zero insulin concentration ($GEZI$) characterizes insulin-independent glucose consumption. V_G represents the apparent glucose distribution volume, expressed in deciliters (dL). The gradual assimilation of meals into the system is governed by the time constants τ_{D_i} .

I enhanced the Identifiable Virtual Patient (IVP) model following the approach outlined in [R155]. This enhancement involved incorporating a Continuous Glucose Monitoring (CGM) error model into the IVP, which accounts for various factors including temporal delay [R156], additive sensor noise, and calibration imprecision. However, to prevent interference with the training process, I opted to exclude sensor drift from the model.

In the extended model, the temporal delay characteristic of CGM measurements, stemming from their interstitial measurement site, was integrated by introducing an additional interstitial glucose compartment denoted as I_G . The sensor noise was modeled as a second-order autoregressive process with a stochastic white noise term w , distributed as $\mathcal{N}(0, \sigma^2)$.

$$\dot{I}_G(t) = -\frac{1}{\tau_{IG}} I_G(t) + \frac{1}{\tau_{IG}} G(t), \quad (3.23)$$

$$v(t) = \alpha_1 v(t - T_s) + \alpha_2 v(t - 2T_s) + w(t), \quad (3.24)$$

$$CGM(t) = I_G(t) + v(t), \quad (3.25)$$

3.5.4 Reinforcement Learning

Utilizing a closed-loop control scenario enables adaptation to the requirements of the target system through semi-supervised learning methods. In the realm of reinforcement learning, the core components consist of the agent, responsible for decision-making, and the environment, which interprets and executes the agent's actions. Over time, the agent learns to select appropriate actions from the environment's action space, aiming to maximize cumulative rewards. The environment provides feedback in the form of a new state and a reward for each action taken by the agent. While the reward aspect is supervised, allowing for precise estimation of state quality,

the exploration of action sequences remains unsupervised, as the optimal sequence of actions is unknown [R157].

I employed the actor-critic reinforcement learning architecture, which combines elements of both policy-based and value-based models [R158]. In reinforcement learning, the agent explores various actions in different states to identify the optimal action sequence that maximizes the reward value. Actor-critic models consist of two key components: the actor and the critic. The actor learns the policy by receiving feedback from the critic, while the critic learns the value function, which assesses the quality of each state. Actor-critic models typically feature two neural networks: one for outputting the value function and another for determining the policy. The value network estimates the goodness value of a state, derived from the value function, while the policy network outputs a vector or scalar with probabilistic values representing the available actions [R158].

Proximal Policy Optimization (PPO) is a policy gradient method utilized in reinforcement learning [R159]. Recent years have seen significant research into policy gradient methods, with algorithms like TRPO, GAE, and A2C/A3C demonstrating superior performance compared to traditional methods such as Q-learning. Among these, the Proximal Policy Optimization Algorithm, developed by OpenAI, stands out as a core algorithm in the policy gradient/actor-critic domain.

In my study, I utilized the PPO model class from the Stable Baselines3 library [R160]. This library offers ease of use across various reinforcement learning algorithms and facilitates the creation of custom environments. All environments are constructed using the OpenAI Gym [R161], ensuring straightforward setup of reinforcement learning environments. Moreover, the library package undergoes regular maintenance, with the swift introduction of new reinforcement learning algorithms. I am particularly interested in testing the TQC algorithm [R162] in the future, which is already available in the Stable Baselines3 contribution repository.

3.5.5 Closed Loop

In my approach, the primary focus lies on the Identifiable Virtual Patient (IVP) model, while the RL agent assumes the role of the closed-loop controller, responsible for administering insulin dosages. The IVP model dynamically predicts blood glucose (BG) levels based on various inputs, including carbohydrates and insulin. The RL agent calculates the insulin dosage for the upcoming time step (5 minutes) by considering current BG levels and the historical record of administered insulin. This results in a closed-loop system, as illustrated in Figure 3.29, and further elaborated through equations (3.26) to (3.29).

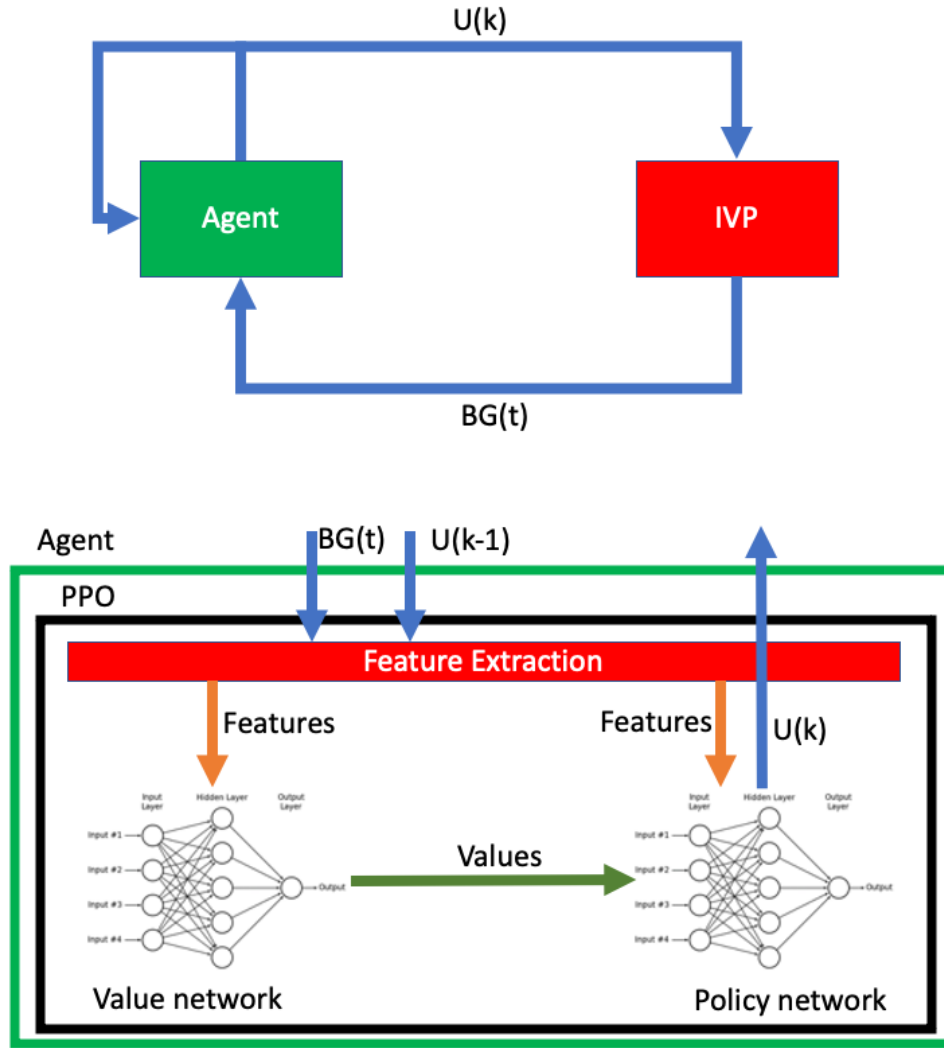


Figure 3.29: The top figure represents the block diagram of the reinforcement learning based closed-loop algorithm. The bottom figure details the control architecture of the system.

3.5.6 Meal generator

I implemented the meal consumption generation scheme outlined in [R163], which incorporates six meals throughout the day. These meals vary in terms of occurrence, carbohydrate content, and timing, based on predefined probabilities.

- Assume that the patient's body weight is known
- Probability of meal appearance: $p = [0.95, 0.3, 0.95, 0.3, 0.95, 0.3]$
- Meal time upper boundary: $up = [9, 10, 14, 16, 20, 23] \cdot 60$
- Meal time lower boundary: $lo = [5, 9, 10, 14, 16, 20] \cdot 60$
- Meal time: $\mu_t = [7, 9.5, 12, 15, 18, 21.5] \cdot 60$
- Meal time variance: $\sigma_t = [60, 30, 60, 30, 60, 30]$
- Amount of meal: $\mu_a = [0.7, 0.15, 1.1, 0.15, 1.25, 0.15] \cdot BW$
- Amount of meal variance : $\sigma_a = \mu_a \cdot 0.15$

- Meal set: $E = \emptyset$
- for $k \in [1, 2, 3, 4, 5, 6]$ do
- Generate a random value for p_{tmp} between 0–100
- if $p_{tmp} \leq p[k]$ then
- Calculate the meal amount for k-th timestamp: $e_k = Round(max(0, Normal(\mu_a[k], \sigma_a[k])))$
- Calculate the meal time for k-th timestamp: $\zeta_k = Round(TruncNorm(\mu_t[k], \sigma_t[k], lb[k], op[k]))$
- Make a union with the meal set: $E \cup \{e_k, \zeta_k\}$
- return E

This method enables the generation of patient-specific carbohydrate intake for a day. Ten-day-long scenarios were created by concatenating ten individually generated day-long samples. The method incorporates randomization, allowing for variation in the number of meals per day (ranging from 2 to 6) as well as in the carbohydrate content and meal timing. Larger meals are constrained to have their upper and lower limits set two hours apart, while other meals may vary by up to half an hour.

Every 5 minutes, the agent determines the insulin dosage to administer to the user. Each simulation begins at 0:00 and spans 24 hours. At each time step, my neural network model receives a feature vector computed from the preceding 12 steps. Training the model involved 5 million episodes, comprising 5 million cases generated by the algorithm.

3.5.7 First results

In my first study [C3] the feature vector was derived from blood glucose measurements and insulin administrations spanning a one-hour period. Given the 5-minute measurement interval of the CGM sensor, a total of 12 blood glucose samples were utilized.

During feature extraction, 11 features were generated from the blood glucose data, alongside 11 features from the insulin data. Specifically, the blood glucose data yielded 11 point deviations from the 12 measured points. Similarly, for the insulin data, 11 features were derived using a pointwise variation approach. Notably, adjustments were made for the insulin data to account for the sensor's 5% measurement increment [C3].

Furthermore, the feature extraction process included computing differences between end-points for the insulin characteristics.

$$\Delta G(I) = G(I + 1) - G(I), \quad I \in [0, 10] \quad (3.26)$$

$$\frac{\Delta G(I)}{\Delta t} = (G(I + 1) - G(I))/5, \quad I \in [0, 10] \quad (3.27)$$

$$\Delta u(I) = u(I + 1) - u(I), \quad I \in [0, 10] \quad (3.28)$$

$$\Delta u = u(11) - u(0) \quad (3.29)$$

In the initial investigation, I experimented with various types of agents, encompassing both discrete and continuous action space agents. The algorithms subjected to testing comprised:

- PPO [R164]
- SAC [R165]
- DDPG [R166]

- DQN [R167]
- A2C [R168]
- TD3 [R169]

I established a closed-loop system, incorporating a virtual patient as the environment. My system featured a neural network-based agent as the controller, responsible for administering insulin into the environment, thereby influencing it. The environment, in response, provided a blood glucose value based on the injected insulin, predetermined patient parameters, and specified carbohydrate intake. The action space ranged from 0 to 18 U/hr for insulin administration.

I investigated different types of reward functions to optimize the controller and found that a two-step reward function was the most effective for my specific problem. Initially, I computed a temporal reward, followed by averaging several temporal rewards in the second step. Evaluation of the tests primarily relied on Time in Range and CVGA metrics. Given the superior performance of the PPO algorithm in the initial tests, I proceeded to employ it for the one-day test analysis. For the day test, I established three distinct test cases.

First test case

In my initial trial, I conducted a one-day simulation for educational purposes, followed by a one-day assessment phase. Among the patients, 5 exhibited significant fluctuations in glucose levels, prompting me to halt the testing protocol. For the remaining patients, there were no noteworthy deviations observed in their glycemia patterns. The average Time in Range (TIR) recorded was 77.55%, a satisfactory outcome considering the broader patient cohort. The distribution of blood glucose (BG) levels across all patients also correlates well with the TIR metric.

Assessing the results using the Coefficient of Variation of the Glucose Area (CVGA) metric, I identified zone B control actions for a total of 7 patients. Notably, patient number 7 demonstrated exceptional performance, maintaining a flawless 100% adherence within the target range.

Second test case

For the subsequent evaluation, I initiated a one-day training phase for the agent, followed by an extensive 10-day simulation period. The aim was to assess the agent's ability to generalize from a single day of training to an extended simulation. Throughout this trial, I encountered 4 instances where extreme conditions necessitated the termination of testing. Unlike the previous trial, I observed noticeable deviations in the agent's performance outside the middle range, resulting in elevated blood glucose (BG) levels.

Interestingly, only two instances of zone B control were apparent in this trial. Regarding the Time in Range (TIR) metrics, the average stood at 72%. Particularly noteworthy was the performance of patient 10, exhibiting exceptional results. An intriguing scenario unfolded with patient 13, where the agent successfully steered the patient away from extreme conditions while leaning towards higher BG levels within the non-extreme zone.

The 180-250 BG range was occupied, on average, 17% of the time, comfortably within the specified acceptable threshold (25%). This consistency with the TIR metric validates its reliability. However, for patient 13, the allowed percentage was exceeded, prompting considerations regarding the agent's general applicability.

Third test case

For my third evaluation, I reversed the conditions of the second test: the agent underwent a ten-day training phase, followed by a condensed one-day testing period. The objective was to determine whether the agent's ability to generalize from extensive training could be applied to shorter testing scenarios.

Unfortunately, the outcomes of this test do not support the agent’s proficiency in formulating effective control strategies for individual patients to avoid extreme zones: the Time in Range (TIR) metrics fail to meet the predefined criteria, as blood glucose (BG) levels displayed shifts towards the higher range.

These findings imply that consistent results might be achievable if I maintain the same simulation duration for both training and testing phases. Additionally, it’s worth noting that no CGM noise was applied to the patients in this particular test.

3.5.8 Second results

In the second study [C2], my aim was to explore the efficacy of various reward functions, particularly considering the absence of continuous functions in previously published tests. The mathematical model underwent modification for this experiment, with the inclusion of CGM noise. Both training and testing simulations were conducted over a one-day period.

During training, carbohydrate inputs were randomly generated, while exercise-induced carbohydrate intake followed the guidelines outlined in the referenced meal section. Thirteen distinct test day carbohydrate intake patterns were defined: 10 test days featured randomly generated carbohydrate intake, while 3 test days had predefined carbohydrate intake scenarios. These included no carbohydrate intake, a consistent 12-gram intake per hour, and an additional 5 grams of carbohydrate introduced at the 12th hour, reaching the maximum carbohydrate intake capacity of the generator.

For control, I utilized the PPO algorithm, chosen for its superior performance in my initial tests. The architectural design was relatively straightforward, with both actor and critic networks structured as outgoing networks. They shared an identical structure, comprising two hidden layers with a ReLU activation function in each layer. Additionally, the observation space was altered, consisting of only two elements representing blood glucose and insulin concentrations at the preceding time point.

I experimented with four distinct reward functions, one of which lacked continuity. All functions were optimized to center around a glucose level of 120 mg/dl. Additionally, I assessed each function under two conditions: halting the simulation if glycemia exceeded a predetermined range, and allowing the simulation to continue indefinitely. I employed the CVGA and Time In Range metrics to evaluate the out-of-range interval in a consistent manner. Furthermore, I computed the mean squared error of the simulated blood glucose during the test, considering both 90 mg/dl and 150 mg/dl thresholds.

Bump function [R170]

$$score(I) = \begin{cases} \exp\left(\frac{-1}{1-((CGM-90)/45-1)^2}\right) & 90 < CGM < 180 \\ 0 & \text{otherwise} \end{cases} \quad (3.30)$$

Piecewise constant function

$$score(I) = \begin{cases} -1 & CGM < 70 \\ 1 & 70 \leq CGM \leq 180, \\ 0 & 180 < CGM \end{cases} \quad (3.31)$$

$$R = \frac{1}{n} \sum_{I=1}^n score(I), \quad (3.32)$$

where R is the final reward and n is the simulation length.

Cosine function

$$score(I) = \begin{cases} -\cos\left(\frac{CGM}{45}\right) & 0 < CGM < 300 \\ -1 & otherwise \end{cases}, \quad (3.33)$$

Mexican hat wavelet [R171]

$$score(I) = \frac{2}{\sqrt{3} \cdot \pi^{\frac{1}{4}}} \cdot \left(1 - \left(\frac{CGM - 140}{140}\right)^2\right) \cdot \exp\left(-\left(\frac{1}{2} \cdot \left(\frac{CGM - 140}{140}\right)^2\right)\right) \quad (3.34)$$

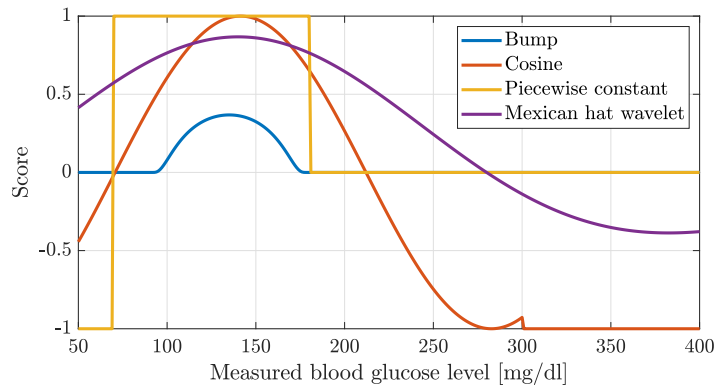


Figure 3.30: Investigated reward functions.

3.5.9 Third Results

In my third experiment [C1], I investigated how altering hyperparameters affects performance. The aim was to identify which hyperparameters significantly impact the control problem. To this end, I conducted tests varying the number of neurons in the hidden layers between 64 and 512. Additionally, I performed another test where the number of neurons in the hidden layers was fixed at 64, but I changed the activation functions (sigmoid, ReLU, ELU, and none). The experimental setup remained consistent with my second experiment, featuring two characteristic inputs and one output representing the insulin quantity.

ReLU:

$$f(x) = \max(0, x) \quad (3.35)$$

Sigmoid:

$$f(x) = \frac{1}{1 + e^{-x}} \quad (3.36)$$

ELU:

$$f(x) = \begin{cases} x & x \geq 0 \\ \alpha(\exp(x) - 1) & x < 0 \end{cases} \quad (3.37)$$

The training and testing scenarios remained consistent with those employed in the second trial. Additionally, I maintained the fixed reward function (piecewise), as originally utilized in

my investigation, and summarized the outcomes using median values. Surprisingly, altering the number of neurons did not yield substantial changes in the control capabilities of the neural network models, despite slightly better performance observed in larger meshes (the 512-neuron network exhibited the most optimal performance in the trials). Conversely, varying activation functions led to significantly divergent controller performance, with ELU and sigmoid functions surpassing others. Notably, the ELU activation achieved a TIR metric exceeding 70%. These findings suggest that employing deeper meshes and ELU activation functions in the hidden layer is advisable.

3.5.10 Final neural network

In the [J4] adaptation of the neural network Figure 3.31, I employed the bump function as my reward function. Consistent with my previous experiments, my input feature vector comprised two elements: the blood glucose level from the previous time step and the quantity of insulin administered at the previous time step. To normalize these features to values between 0 and 1, I divided the blood glucose level by 1000. For insulin, as the action space in the environment ranged from -1 to 1 (representing values from 0 to 25 U/hr), I adjusted the feature vector design accordingly. I added one to the current decision and then divided by 2. Consequently, when the network made a decision, I added 1 to that value, multiplied it by 25, and finally divided by 2 to obtain the actual insulin value to be administered to the patient. Building on my earlier findings that models perform better without halting simulation when values exceed limits, I simulated the entire time period regardless. Given that extended simulation time yielded more accurate results, I opted for a simulation duration of 10 days based on my test outcomes.

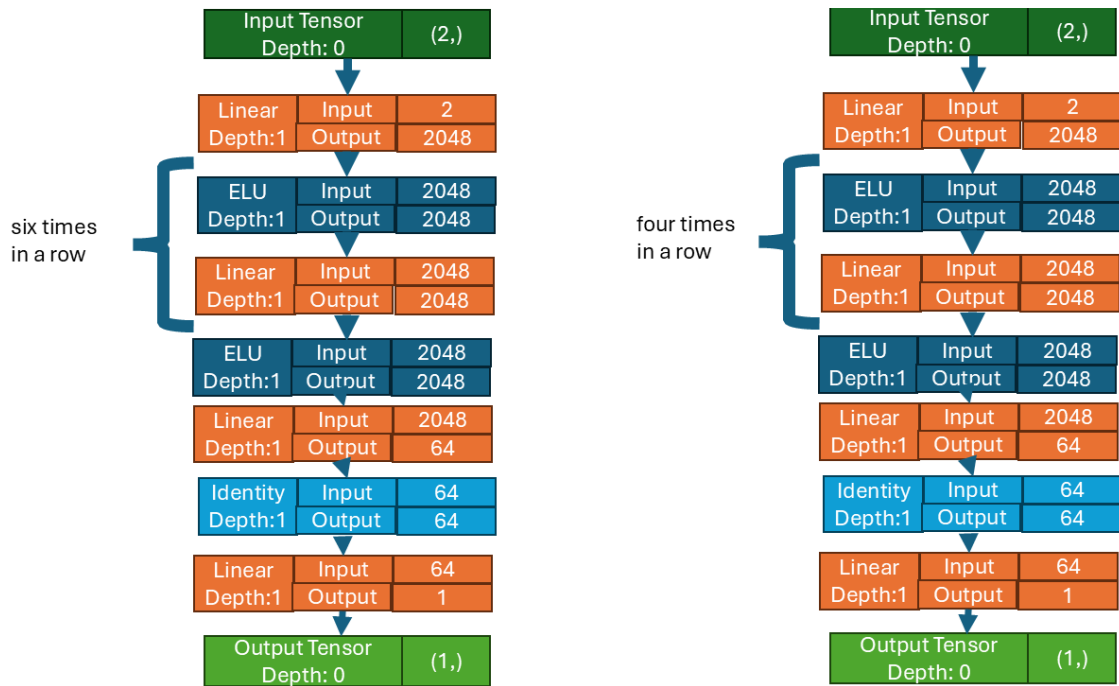


Figure 3.31: A graphical representation of the policy and value networks as depicted by Pytorch. The image on the left is the policy network, which is larger than the value network. This network learns the control strategy. On the right is the value network, which gives an estimate of how good the policy network was.

My current network retained the distinction between value and policy, a structure consistent with my previous tests. To avoid shallow networks and incorporate larger networks, I introduced blocks comprising Linear and ELU layers in my current experiment. The inclusion of the ELU

layer was motivated by the favorable outcomes observed in the third test, while the incorporation of the Linear layer was driven by my continuous output requirement. The policy and value networks featured a variable number of these blocks: the value network comprised 5 blocks, while the policy network contained 7 blocks. I opted for a smaller value network to facilitate quicker learning of the critical aspects of my model, whereas the policy network was designed deeper to provide the actor with greater flexibility in learning the optimal strategy. Each block typically consisted of 2048 neurons, except for the initial and terminal blocks. The initial block in both networks had 2 features in its input layer, while the terminal block had an output of 64 values. Both networks concluded with a terminal block composed of an Identity layer and a Linear layer. The output of each network was a single value: for the value network, this value represented a prediction of the quality of the given step, while for the policy network, the value, ranging between -1 and 1, indicated the quantity of insulin to be administered. The network architecture is illustrated in Figure 3.32.

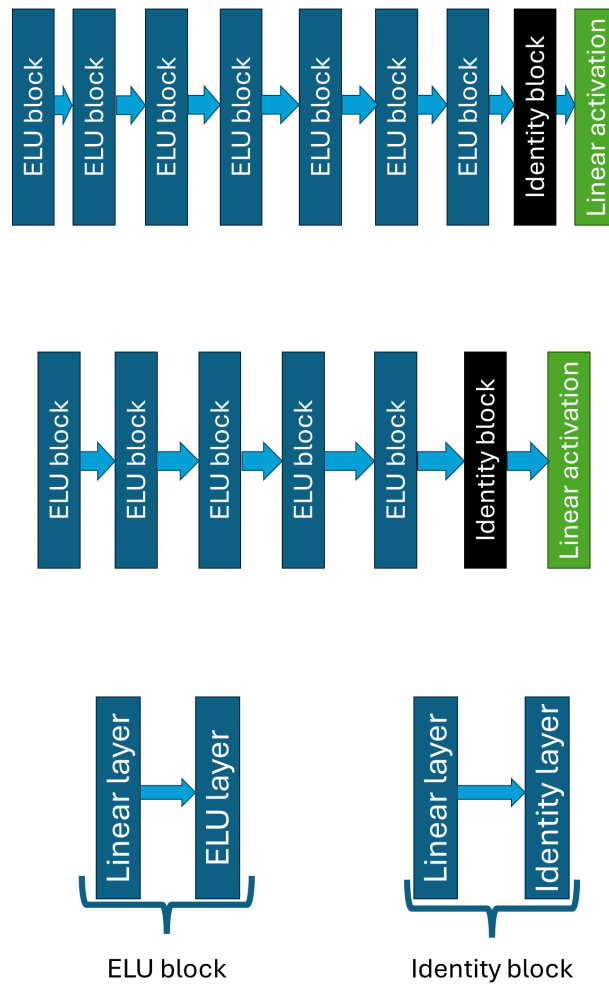


Figure 3.32: The top figure represents the architecture of the Policy Network, which consists of multiple Exponential Linear Unit (ELU) blocks followed by an Identity block and a final Linear activation layer. The middle figure shows the architecture of the Value Network, which is similarly structured with a sequence of ELU blocks, an Identity block, and a Linear activation layer at the output. The bottom figure illustrates the internal structure of the ELU and Identity blocks, where the ELU block consists of a linear layer followed by an ELU activation, and the Identity block replaces the ELU with an identity activation function after the linear layer.

In my third trial [C1], I explored the impact of adjusting hyperparameters on performance.

My objective was to identify which hyperparameters significantly influence the control problem. To achieve this, I conducted experiments altering the number of neurons in the hidden layers within the range of 64 to 512. Additionally, I conducted another trial with a fixed number of neurons (64) in the hidden layers but varied the activation functions (sigmoid, ReLU, ELU, and none). The experimental setup remained consistent with my second experiment, involving two characteristic inputs and one output representing the insulin quantity.

The training and testing scenarios remained consistent with those employed in the second trial. Additionally, I maintained the fixed reward function (piecewise), as originally utilized in my investigation, and summarized the outcomes using median values. Surprisingly, altering the number of neurons did not yield substantial changes in the control capabilities of the neural network models, despite slightly better performance observed in larger meshes (the 512-neuron network exhibited the most optimal performance in the trials). Conversely, varying activation functions led to significantly divergent controller performance, with ELU and sigmoid functions surpassing others. Notably, the ELU activation achieved a TIR metric exceeding 70%. These findings suggest that employing deeper meshes and ELU activation functions in the hidden layer is advisable.

In the adaptation of the neural network, I employed the bump function as my reward function. Consistent with my previous experiments, my input feature vector comprised two elements: the blood glucose level from the previous time step and the quantity of insulin administered at the previous time step. To normalize these features to values between 0 and 1, I divided the blood glucose level by 1000. For insulin, as the action space in the environment ranged from -1 to 1 (representing values from 0 to 25 U/hr), I adjusted the feature vector design accordingly. I added one to the current decision and then divided by 2. Consequently, when the network made a decision, I added 1 to that value, multiplied it by 25, and finally divided by 2 to obtain the actual insulin value to be administered to the patient. Building on my earlier findings that models perform better without halting simulation when values exceed limits, I simulated the entire time period regardless. Given that extended simulation time yielded more accurate results, I opted for a simulation duration of 10 days based on my test outcomes.

My current network Fig. 3.31 retained the distinction between value and policy, a structure consistent with my previous tests. To avoid shallow networks and incorporate larger networks, I introduced blocks comprising Linear and ELU layers in my current experiment. The inclusion of the ELU layer was motivated by the favorable outcomes observed in the third test, while the incorporation of the Linear layer was driven by my continuous output requirement. The policy and value networks featured a variable number of these blocks: the value network comprised 5 blocks, while the policy network contained 7 blocks. I opted for a smaller value network to facilitate quicker learning of the critical aspects of my model, whereas the policy network was designed deeper to provide the actor with greater flexibility in learning the optimal strategy. Each block typically consisted of 2048 neurons, except for the initial and terminal blocks. The initial block in both networks had 2 features in its input layer, while the terminal block had an output of 64 values. Both networks concluded with a terminal block composed of an Identity layer and a Linear layer. The output of each network was a single value: for the value network, this value represented a prediction of the quality of the given step, while for the policy network, the value, ranging between -1 and 1, indicated the quantity of insulin to be administered. The network architecture is illustrated in Figure 3.32.

3.5.11 Training Phase

During the iterative training phase, a sequence of 10 consecutive-day scenarios served as the test set to evaluate the evolving performance of the neural network. These scenarios included randomized meal patterns interspersed between one-day periods, providing diverse patterns for the networks to learn from. Multiple networks were trained for each subject, maintaining constant patient parameters but varying the timing and carbohydrate content of meals. Training

was completed over 5 million iterative steps and conducted on the ELKH research cloud.

3.5.12 Testing Phase

The evaluation phase involved 13 distinct meal schemes, encompassing both random and intentionally designed scenarios to subject the classification algorithms to a broad spectrum of challenges. The inclusion of human-designed scenarios aimed to assess potential overfitting. Virtual patient parameters remained constant throughout both the training and testing phases.

Among the 13 test scenarios, 10 replicated the methodology utilized during the training phase, while three were manually crafted to represent specific situations: the first scenario featured no meal intake (0 g); the second scenario involved administering 12 g of carbohydrates every hour; and in the third scenario, a single substantial meal with maximum carbohydrate content was administered, with an additional 5 grams of carbohydrates added.

3.5.13 Result and Discussion

Table 3.13 summarizes the aggregated results across different glucose ranges. In the 70-180 mg/dL range, the median value is notably high at 86%, exceeding previous results. The method achieved better than 44% Time in Range (TIR) in 75% of the tests, though the lowest recorded value in this range was 18%, indicating that some patients did not achieve optimal glucose control with this method. Notably, outliers in the results were patients with low body weight. Overall, the average TIR exceeded 70%, representing an improvement over my earlier studies.

In the 180-250 mg/dL range, over 75% of patients maintained glucose levels below 180 mg/dL during the evaluation period. Similar patterns are evident in the >250 mg/dL range, suggesting that elevated blood glucose levels occurred only in isolated cases, all within the permissible 5% limit.

The RMSE150 variable represents the root mean squared error from a glucose concentration of 150 mg/dL. Half of the data deviated by less than 37 mg/dL from this value, and for the upper quartile, a deviation of 65 mg/dL is a reasonable outcome. In fact, a quarter of the test values ranged between 85 mg/dL and 215 mg/dL.

Table 3.13: Aggregated table for all investigated simulation days, including mean, maximum, minimum, standard deviation, and quarterlies values in terms of TIR zones.

	<50	50–70	70–180	180–250	>250	RMSE90	RMSE150
mean	11.925	14.013	73.261	0.512	0.289	94.777	69.905
std	21.647	17.288	28.350	1.482	1.466	79.771	67.590
min	0.000	0.000	18.056	0.000	0.000	20.016	17.106
25%	0.000	0.000	44.792	0.000	0.000	43.382	31.384
50%	0.000	8.333	86.806	0.000	0.000	61.408	37.472
75%	11.458	22.222	100.000	0.000	0.000	109.674	65.366
max	73.264	73.264	100.000	9.375	11.458	360.283	309.865

Figure 3.33 displays the median and standard deviation for the 221 days simulated during the test. The median blood glucose values were calculated for this simulation and are presented on two subbars: one showing only the median value, and the other showing the median alongside values between the 10th and 90th percentiles. Carbohydrate intake is also plotted. The median value falls within the desired range, indicating satisfactory performance. Notably, the 10th percentile values also remain within this range, which is encouraging. However, some patients, particularly those with lower body weight, had simulated blood glucose values reaching as high

as 400 mg/dL, highlighting the need to improve the system’s robustness an area I will focus on in future studies.

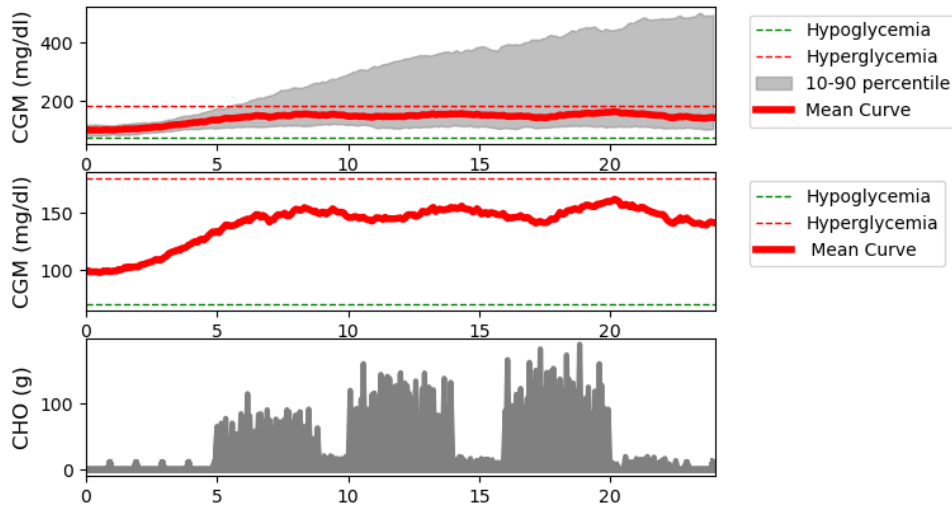


Figure 3.33: Blood glucose median curve with standard deviations and meal intakes for all the investigated days.

Figure 3.34 is similar to Figure 3.33, but it plots the mean values of the data simulated during the test. Instead of percentile differences, the standard deviation of ± 2 is shown. It is worth noting that the mean values remained largely within the target range for most of the simulation but began to drift out of range toward the end. There is a noticeable steady increase in the blood glucose curve, which can be attributed to instances where the neural network model failed to administer insulin, leading to a continuous rise in blood glucose. Since the mean is sensitive to such deviations, the overall simulation tended toward higher blood glucose levels, which is further reflected in the large variance observed. This highlights the insufficient robustness of the RL controller, particularly for patients with lower body weights. However, the small difference between the median and mean blood glucose values is a promising result.

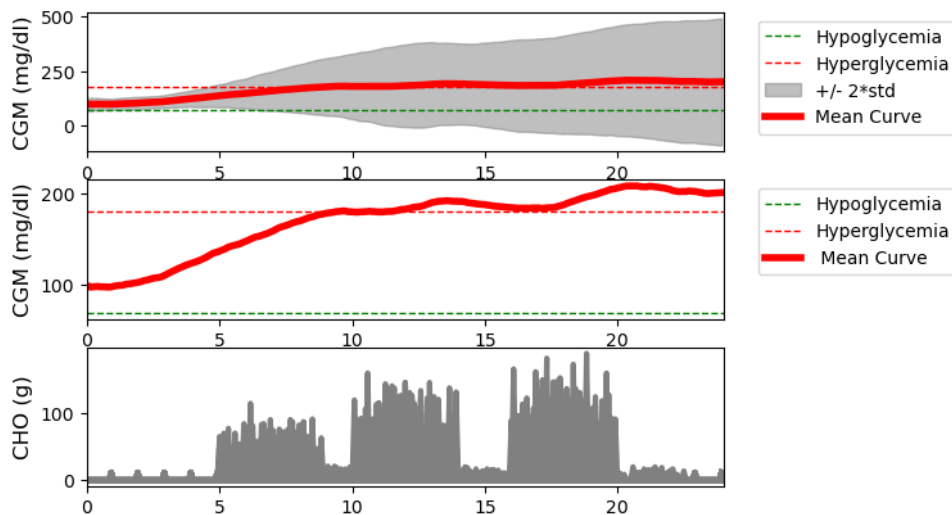


Figure 3.34: Mean blood glucose curve with standard deviation and meal intakes for all the investigated days.

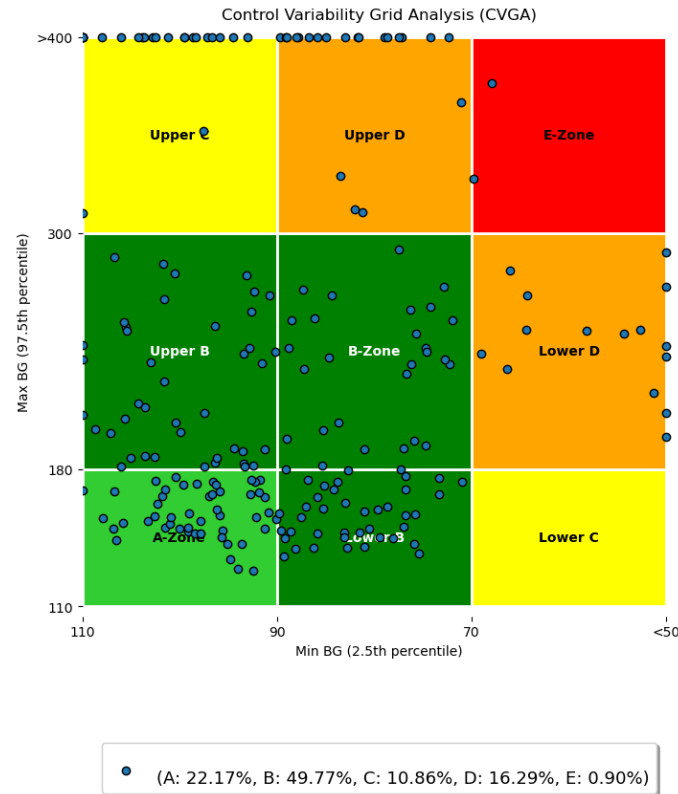


Figure 3.35: CVGA diagram for all tested days

Figure 3.35 illustrates the Control Variability Grid Analysis. A significant proportion of the samples fall within the A and B zones. Compared to my previous study, where Zone A comprised approximately 1% of the samples, the current method achieves a substantial improvement with 22% of the samples falling within Zone A. Similarly, there is an increase in the occurrences of Zone B, rising from around 30% to almost 50%. Zone C occurrences have remained consistent with previous levels. Samples in Zone Upper D have decreased significantly, approximately halving, while Zone E samples have been minimized.

Noteworthy spikes are observed in both the high and low blood sugar level sections. Interestingly, there is a higher concentration of test cases in the higher blood glucose half at each extreme, indicating a tendency towards hyperglycemia. Conversely, there are fewer points in the low blood glucose section, suggesting that the model effectively avoids hypoglycemia. However, it is evident that some patients still experience hyperglycemia.

After presenting my results, I now delve deeper into the outcomes. Starting with Table 3.13, I see the aggregated results for all simulated days. For glucose levels below 50 mg/dL, half of the simulated days had a value of 0, indicating that in 50% of cases, there were no significant hypoglycemic events. However, for the 75th percentile, the value is 11, meaning that 25% of the simulated days experienced significant hypoglycemia. Specifically, there were 55 days when blood glucose levels dropped below 50 mg/dL during the simulation. Even more concerning, the maximum value in this category reached 73, highlighting that there are patients for whom my model is not functioning effectively. These cases mainly involved patients with lower body weight, where the model administered a large dose of insulin, leading to dangerously low blood glucose levels and an inability to return to a healthy range. To avoid this, insulin should not be administered when glucose levels are already low. In the 50-70 mg/dL range, at least half of the simulations show values below 9, while a quarter of the simulated days fall outside this range entirely. The value in the 75th percentile is double that of the median, indicating that many patients had blood glucose levels within this range during the simulation. According to Time

in Range (TIR) metrics, time spent in this zone should not exceed 5 minutes, though values between 50-70 mg/dL are not considered a serious problem if they don't drop below 50 mg/dL. The most important column is the TIR metric, where it is promising that more than 50% of simulated days achieved TIR values greater than 70% in the 70-180 mg/dL range. In fact, the 75th percentile shows that 25% of the days stayed 100% within this range. The average TIR value exceeded 70%, a result not observed in previous tests. However, 25% of the simulated days had TIR values under 40%. The next two columns, which represent higher blood glucose levels, show minimal values compared to previous studies, indicating that the model generally avoided high glucose levels by not overdosing insulin. However, on certain days, particularly with lightweight patients, the model reacted too late to sudden blood sugar spikes following carbohydrate intake. Figures 3.33 and 3.34 present simulation results for all days, including the mean and median curves. For the mean, the standard deviation of ± 2 is plotted, while for the median, the 10th to 90th percentiles are shown. We also examined carbohydrate intake, which is clearly reflected in the graphs by the three large peaks representing main meals, along with visible shifts in timing, suggesting that random days were used for testing. For the median curve, the model was able to maintain glucose levels within the TIR zone throughout the day, although at the start, the models allowed blood glucose to rise before stabilizing around 150 mg/dL. Observing the percentiles, it is clear that even the 10th percentile remained within the TIR range, though the 90th percentile reached up to 400 mg/dL, which is excessively high. This indicates that while the model handles hypoglycemia well, it still struggles with hyperglycemia. The average curve, unlike the median, drifted out of the TIR zone by the end of the day, showing a steadily increasing slope, supporting my earlier observation that the simulation time for training was too long. However, the average did not exceed 200 mg/dL. The scatter plot reveals that the maximum value can reach as high as 500 mg/dL, suggesting that my model is ineffective for certain patient cases, particularly those with lower body weight. In these cases, the model either reacts too late or not at all, and administering insulin too early causes significant fluctuations in glucose levels. The CVGA plot in Figure 3.35 shows all test days, and in Zones A and B where control is considered most successful I achieved 72%, meaning nearly three quarters of the days demonstrated good control. However, many days shifted into the upper range for poor control, as evidenced by more data points in Upper C and Upper D than in Lower C and Lower D. In Zone E, which indicates complete loss of control, I only observed 1% of the days, meaning just one or two days were borderline failures. Overall, my solution worked well for most days, but when it failed, it usually resulted in high blood glucose levels rather than low ones.

3.5.14 Conclusion

When employing reinforcement learning for glycemic control and developing a new agent model, I successfully increased the median Time in Range (TIR) metric value to around 80%. Moreover, there was a notable increase in the number of samples falling within Zone A and Zone B in the CVGA analysis. Compared to previous studies, this novel approach generally reduces glycemic curves, resulting in significantly improved TIR values, albeit with a slight increase in Time Below Range (TBR) values.

Instances of hypoglycemia were primarily observed in patients weighing less than 65 kilograms, indicating a need to develop effective and safe control methods tailored to individuals with lower body weight. Addressing simulation time is also crucial and should be minimized in future studies. Additionally, attention should be directed towards refining the policy algorithm, as well as addressing instances where the Proximal Policy Optimization (PPO) model may become trapped in local minima during testing. Furthermore, exploring learning with a larger feature vector may resolve issues related to administering excessive doses of insulin at once.

In future endeavors, I propose conducting two training phases. The first phase would involve fixed time carbohydrate intakes, allowing the agent to learn how to manage carbohydrate intake of varying sizes. Subsequently, in the second phase, the timing of carbohydrate intake would

be varied. This sequential approach aims to streamline the optimization process for the agent, focusing on one aspect at a time rather than multiple variables concurrently.

3.6 Thesis Summary – Diabetes Research

This chapter presents my research contributions in the context of diabetes management, focusing on three interconnected areas: physical activity detection, gesture (meal) detection, and insulin regulation using artificial intelligence techniques. Although structured as a single thematic unit, the research contains three distinct methodological directions. Each part reflects novel approaches and experimental results that advance current practices in diabetes technology.

Physical Activity Detection in Diabetes Patients

The first and most extensively developed topic addresses the detection of physical activity in diabetes patients using noninvasive signals. I investigated whether physical activity can be identified using only heart rate and blood glucose data—without requiring dedicated motion sensors.

As a novel approach, I showed in [J1] that even traditional machine learning models (e.g., decision trees, SVM) could detect physical activity with acceptable accuracy, demonstrating the feasibility of using physiological signals as proxies for movement. Later, I significantly improved upon these results by applying recurrent neural networks (RNNs), achieving over 90% accuracy as detailed in [J2]. These results confirmed that temporal modeling of physiological signals is highly effective for activity recognition.

Another key contribution was my comparison between simulated and real world datasets [C5], where I demonstrated that simulator based training does not mask realistic patterns and is valid for early stage model development. This was an important validation for the use of synthetic data in medical applications, where patient level data is often scarce or sensitive.

Gesture Detection from Wrist-Worn Sensors

The second topic focused on gesture detection specifically, the recognition of meal related movements from wrist mounted inertial sensors. My initial work involved developing a simple machine learning model trained on accelerometer data collected via wearable devices. This model achieved an accuracy of around 90% [C4], indicating the potential for AI-driven meal detection in real-world scenarios.

The novelty in this part came with scaling the problem to a much larger, publicly available dataset comprising data from more than 300 individuals. I applied recurrent neural networks to this dataset and achieved a classification accuracy of approximately 98% [J3]. A key innovation was the personalized model structure: each patient’s data was used to train a dedicated model, making the detection highly patient specific. I demonstrated that even one day of data was sufficient to train a model with high accuracy, although performance improves further with additional training data. These results suggest that personalized gesture recognition is not only feasible but scalable across diverse patient populations.

Insulin Regulation with Reinforcement Learning

The third part of my research focused on the use of reinforcement learning (RL) to optimize insulin delivery in a personalized, autonomous way. The goal was to replace fixed medical protocols with adaptive, AI-driven control strategies.

In [C3], I tested various RL algorithms and found that the Proximal Policy Optimization (PPO) algorithm consistently produced the most stable and accurate insulin regulation results. A key finding was that model performance degrades when evaluated over time windows longer

than the training period highlighting the importance of training duration in medical control applications.

I then introduced a novel reward shaping strategy in [C2], where I compared different reward functions. I showed that the "bump" reward function outperformed all other alternatives in terms of blood glucose control metrics, including Time-in-Range (TIR) and Mean Absolute Error (MAE). The "cosine" reward also showed potential, but was consistently less effective. Another novelty was demonstrating that uninterrupted training simulating full daily cycles without resetting the environment yields superior learning outcomes.

In [C1], I evaluated the sensitivity of PPO models to various hyperparameters. Notably, I showed that the choice of activation function significantly impacts performance: using the ELU activation function was critical for achieving TIR values above 70%. In contrast, the number of neurons had a minimal effect, while deeper networks offered slight improvements.

Finally, in [J4], I introduced a novel dual-network architecture where the policy and value networks were structurally distinct for the first time. I also extended the training setup by increasing both the number of simulated episodes and the number of steps per episode. These innovations led to the best performance achieved in this line of research. The patient specific models developed in this work achieved high TIR scores and showed generalizability to individuals over 70 kg in body weight. However, the model's performance declined for lighter patients, indicating the need for tailored training based on patient characteristics. Additionally, the experiments confirmed that one day of training is insufficient and that longer simulation periods significantly enhance learning stability.

4. Processing MRI images

Thesis group II: Researches focused on the advanced processing techniques applied to MRI imaging data

Thesis II.1

I developed an artificial intelligence-based solution for the pixel-level segmentation of infant brain tissues from MRI images. My solution can greatly help the medical staff in the detection of abnormal development of the brain.

Thesis II.1.1

I have shown that it is not mandatory to follow the traditional U-net architecture. My experiments revealed that decreasing the filter numbers in consecutive encoding blocks and increasing them in consecutive decoding blocks can also provide fine segmentation.

Thesis II.1.2

I have shown that using non-traditional cost functions in training can be beneficial. Using a Dice similarity score-based cost function gives much better results than using a traditional categorical cross-entropy function.

Thesis II.1.3

I have shown that it is worthwhile to deploy (2+1)D convolution, frequently used in video processing, in the segmentation problem. Consecutive slices of a volumetric MRI record correlate as much as consecutive frames of a video.

Thesis II.2

I have developed solutions that classify brain MRI tumor images more accurately than state-of-the-art models. In addition, I have developed a solution that can classify and segment the tumor simultaneously.

Thesis II.2.1

I have shown that reducing image size to a certain extent improves the classification accuracy of brain tumor MRI images. Furthermore, I have shown that a better training strategy is to save the model continuously when the accuracy on the validation dataset increases during training. I also proved that a large network size is not needed for all problems. In particular, if the problem is specific, smaller architectures may solve the problem better.

Thesis II.2.2

I proved the theory that if you train a network on two problems simultaneously, you can get better results than if you train on only one problem at a time. In addition, I have created a network model that can segment and classify accurately at the same time, when all data is available.

Publications relevant to the theses: [C7, C8, J5, J6, J7].

Medical imaging plays a central role in modern healthcare, enabling clinicians to non-invasively examine internal structures of the human body. Among the various imaging modalities, *magnetic resonance imaging* (MRI) is one of the most versatile and information-rich technologies, offering exceptional spatial resolution, soft tissue contrast, and flexibility in visualizing anatomical and functional aspects of the brain and other organs.

This chapter presents the results of my research in the field of medical image processing, with a focus on applying advanced deep learning models to MRI data. The goal of this work is to improve diagnostic accuracy and facilitate the development of intelligent systems that support clinical decision-making.

4.1 Chapter Overview

The chapter is divided into two main research tracks:

- **Infant brain tissue segmentation:** Accurate tissue segmentation is essential for early diagnosis and developmental tracking in neonates. I developed multiple U-net-based architectures, ranging from 2D and 3D models to a novel spatiotemporal convolutional network. These networks were trained and evaluated on real clinical datasets to segment cerebrospinal fluid (CSF), gray matter (GM), and white matter (WM) with high precision.
- **Brain tumor classification:** In this section, I propose an end-to-end system for identifying three types of brain tumors—glioma, meningioma, and pituitary—based on T1-weighted MRI scans. A two-stage L-net architecture was introduced, combining a segmentation module based on U-net with a CNN classifier. This modular approach enhances the system’s interpretability and robustness while achieving state-of-the-art performance.

4.2 Novel Contributions

The key novelties and contributions of this chapter include:

- **Architectural innovations:** I introduce a range of custom U-net architectures tailored for different use-cases. In particular, the spatiotemporal U-net combines 2D spatial and 1D temporal convolutions to exploit the sequential nature of multi-slice MRI volumes—a design rarely used in infant brain segmentation.
- **L-net architecture:** The L-net framework is a novel dual-component architecture where segmentation and classification are handled sequentially. The U-net segmentor localizes tumor regions, and the CNN classifier operates directly on the segmented areas, improving classification accuracy and interpretability.
- **Comprehensive evaluation:** All models were evaluated on publicly available datasets and assessed using accuracy, AUC, and F1-score. The segmentation results achieved competitive or superior performance compared to existing benchmarks, and the tumor classification models achieved high precision with minimal false positives.
- **Clinical relevance:** The methods proposed are tailored for integration into real-world diagnostic workflows. Both the segmentation and classification pipelines are optimized for automation and scalability, making them applicable in hospital environments with limited expert resources.

Overall, this chapter highlights how deep learning can be harnessed to improve diagnostic imaging and pave the way for intelligent, data-driven radiology systems. The research presented here contributes not only novel technical solutions but also emphasizes the importance of practical deployment in clinical contexts.

4.3 Infant brain tissue segmentation

4.3.1 Goal

The focus of my research was to segment the MRI recordings of the infant brain, to distinguish the three main tissue types, namely the white matter, the gray matter, and the cerebro-spinal fluid. This task is by no means as simple as for adults. In fact, in the first few months after birth, white matter looks darker than gray matter in T1-weighted MRI. The white matter gradually gets a lighter color and approximately at the age of 6 months, these two tissue types apparently have the same color and thus are difficult to distinguish. To provide a reliable automated solution for this segmentation problem is utmost important, as the doctors wish to study the correlation between the brain's development in the first year of life with the occurrence of certain illnesses.

4.3.2 Introduction

Brain tissue segmentation using MRI data has been a subject of extensive research for decades, as demonstrated by studies such as [R172] and [R173]. However, segmenting infant brain tissues introduces additional complexity. In the early months after birth, the T1 data channel of MRI shows a notable difference: gray matter appears lighter than white matter, which facilitates segmentation. By six months of age, the pixel intensity distributions of these tissue types begin to converge significantly. This intricate overlap is highlighted in research by [R174] and [R175].

The development of automated segmentation methods has primarily focused on improving medical workflows through computational efficiency. In the specific case of infant brain tissue segmentation, computers have a unique ability to identify tissue boundaries that may not be easily visible to the human eye. As a result, these systems are crucial in supporting precise decision-making processes.

Research into automated segmentation techniques for infant brain tissue has been extensive over the past decade. Challenges such as iSeg-2017 and iSeg-2019 [R174, R175] have significantly intensified these efforts. These challenges have been vital by providing carefully annotated training datasets and establishing a standardized benchmarking framework, which has driven advancements in the field.

In the early stages, before the introduction of the iSeg challenges, common methodologies primarily relied on classical machine learning techniques [R176]. Methods such as k -nearest neighbors, binary decision trees, and support vector machines [R177, R178] were foundational to these approaches. Additionally, shape models [R179, R180], as well as unsupervised techniques like level sets [R181, R182] and various c -means clustering models [R183, R184], were often used.

The pivotal rise and development of convolutional neural networks (CNNs) began a few years before the iSeg challenges. Thus, it is unsurprising that many solutions for these challenges and later publications are based on variants of CNNs and deep learning techniques. Notable examples include works by [R185, R186, R187]. Moreover, architectures such as deep residual networks [R188] and U-net networks [R189] have become common in infant brain tissue segmentation, demonstrating the rapid adoption of advanced neural network architectures in the segmentation process.

In this chapter I introduce a new solution for the complex task of infant brain tissue segmentation. My approach involves adapting a U-net network specifically tailored to meet the

unique requirements of this problem. A key aspect of my methodology is the use of the training dataset comprising ten records, which was carefully curated by the iSeg-2017 challenge.

To improve the effectiveness of my approach, I incorporate a single preprocessing step into the workflow. This step acts as a normalization mechanism, aligning the histograms within each data channel of the MRI records. This essential normalization process not only adjusts the pixel intensities for meaningful comparisons but also prepares the data for subsequent neural network processing. The output from the neural network is used as the final segmentation result. To objectively evaluate the quality of the segmentation, this output undergoes detailed statistical analysis, providing an unbiased and quantifiable assessment of the results.

4.3.3 Dataset and Challenge Context

This study is built on the meticulously curated training records from the iSeg-2017 challenge, as described by [R174]. These training records are a crucial resource for developing, validating, and assessing my proposed U-net-based methodologies.

The iSeg-2017 dataset consists of ten distinct records, each providing essential information for my research. Each record contains a multifaceted dataset featuring T1-weighted and T2-weighted MRI data acquired from infant subjects. An automated registration method ensures accurate spatial alignment between these complementary MRI data modalities.

An invaluable part of this dataset is the human expert-crafted ground truth, which annotates pixels corresponding to three main tissue categories: cerebrospinal fluid (CSF), gray matter (GM), and white matter (WM). These annotations serve as the reference standard for evaluating the segmentation performance of my proposed U-net-based methods.

Each volume in the dataset comprises up to 112 transverse cross-sections, with dimensions of 144×192 pixels, capturing the detailed anatomical features of the infant brain tissues. Each pixel represents a cubic region measuring one millimeter in each dimension, thus providing critical information about the three-dimensional structure of these tissues.

The challenge’s central issue becomes apparent when considering the overall segmentation task. As infants reach approximately six months of age, a significant convergence occurs in the intensity distributions of white matter (WM) and gray matter (GM). This convergence results in perceptual similarity, making visual differentiation difficult for humans. This ambiguity increases the complexity of the segmentation task, as the intensity-based distinctions between these two vital tissue types become unclear and intertwined [R174].

By thoroughly exploring this rich and complex dataset, my study aims to uncover patterns in infant brain tissue segmentation. I seek to develop innovative methodologies that effectively address the challenges posed by the overlap between white and gray matter tissues.

4.3.4 Data Preprocessing and Intensity Alignment

In magnetic resonance imaging, intensity inhomogeneity has long been recognized as a potential source of signal distortion and complexity [R190, R191]. This low-frequency noise can create irregularities in pixel intensities across the image, complicating subsequent analysis. However, in my examination of the iSeg-2017 training dataset, careful analysis shows that intensity inhomogeneity is minimal. Therefore, developing compensation strategies for this phenomenon is unnecessary for my specific dataset.

Nevertheless, challenges still arise. I encounter occasional gaps and missing data, appearing as voids in certain pixel positions. Addressing these gaps is crucial, as they can compromise the integrity of later processing and analysis. To resolve this, I apply data augmentation and interpolation techniques to ensure continuity of information and maintain the spatial context within these regions.

Beyond these gaps, the primary focus of my preprocessing effort is on harmonizing the diverse intensity scales among the ten MRI records in the iSeg-2017 training dataset. Variations

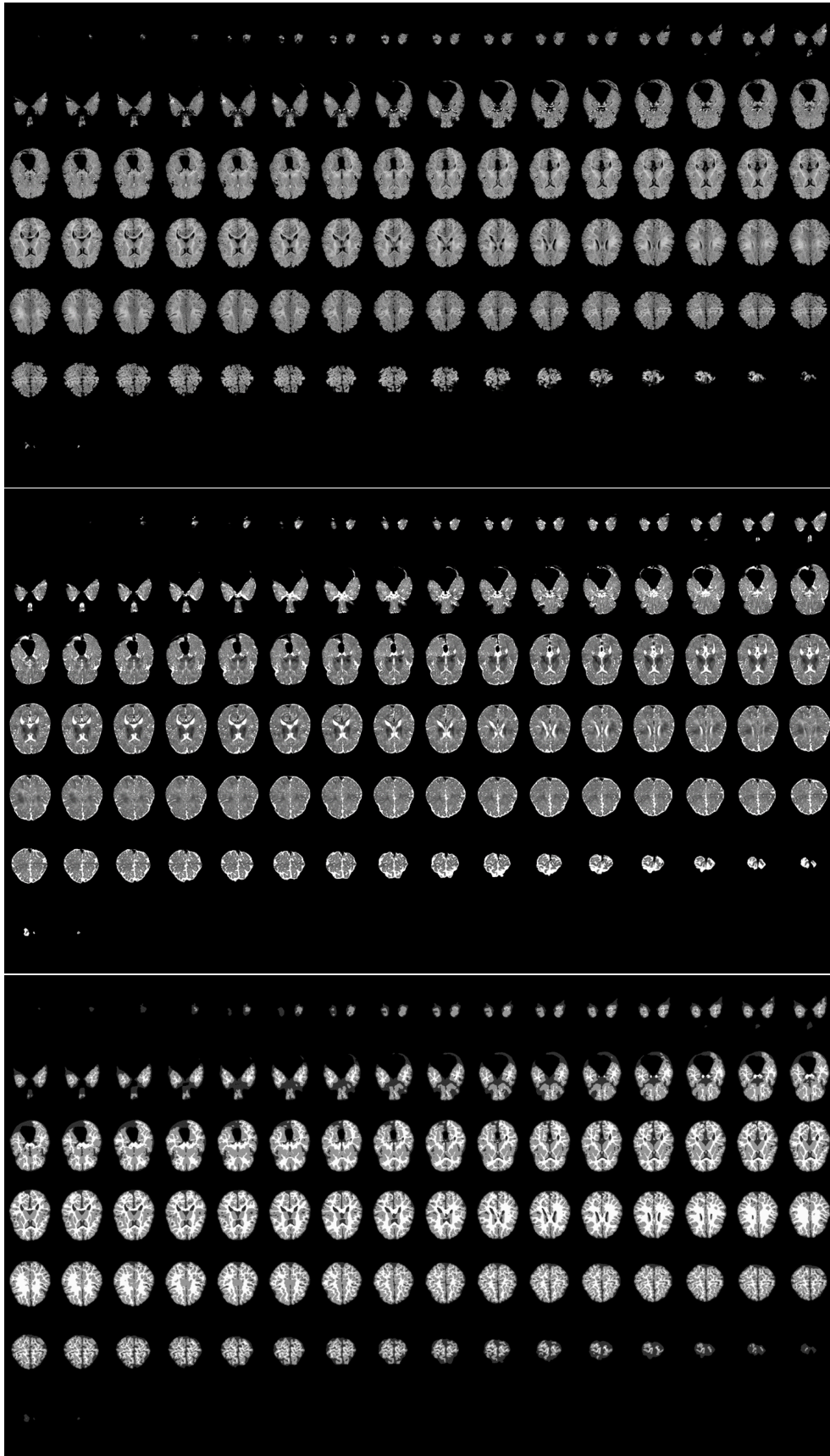


Figure 4.1: A whole volumetric MRI record presented in cross-sections: T1 data in top panel, T2 data in the middle panel, ground truth in the bottom panel.

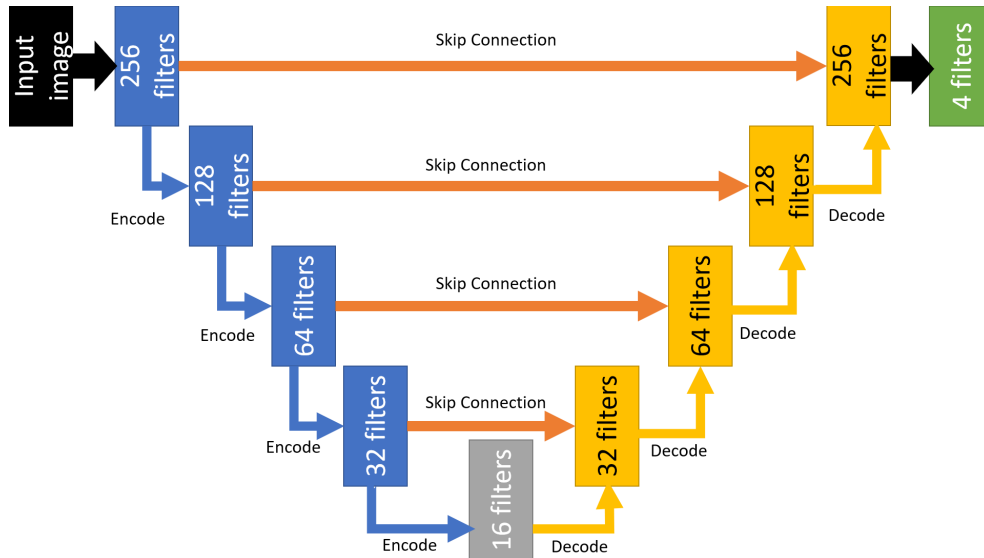


Figure 4.2: The U-net network structure deployed for brain tissue segmentation.

in intensity distributions can originate from differences in acquisition protocols, imaging hardware, or scanning environments. To address these discrepancies, I employ a preprocessing step involving histogram alignment.

For this alignment, I use a method proposed by [R192], which manages intensity scale harmonization by utilizing key percentiles in the intensity distribution. Specifically, I use the 25th, 50th, and 75th percentiles as key markers within the intensity histograms. By aligning these points across the ten MRI records, I create a consistent intensity framework that overcomes the differences caused by varying data acquisition.

My histogram alignment methodology is further refined by insights from previous research, notably presented in my earlier study [R193]. This combination of strategic refinement and established methodology creates a preprocessing approach that not only corrects disparities but also builds on the collective knowledge gained from prior research.

Through these carefully coordinated preprocessing measures, my study aims not just to correct data imperfections and ensure uniformity, but also to establish a strong foundation of data integrity and preprocessing robustness. This foundation, constructed through data augmentation, interpolation, and histogram alignment, prepares the dataset for subsequent segmentation analysis using the U-net neural network architecture.

4.3.5 First U-net network

U-net, introduced by Ronneberger et al. [R194], is a neural network designed for biomedical image segmentation. Today, U-net is widely used for semantic segmentation tasks, primarily because it can effectively learn from a small number of training samples. The network is named for its U-shaped architecture, which consists of an encoder and a decoder section, classifying it as an autoencoder neural network. Typically, both the encoder and decoder sections have four blocks and they are connected by a bridge.

I modified the U-net architecture to suit my specific problem. The network's structure is shown in Fig. 4.2. On the encoder side, the spatial dimensions are progressively halved while the number of filters (features) doubles with each block. Conversely, on the decoder side, the process is reversed: spatial dimensions double, and the number of features is halved. The encoder works as a feature extractor by learning an abstract representation of an image through multiple sequences of encoder blocks. Each block consists of two convolutional layers with a kernel size of 3×3 , followed by a Rectified Linear Unit (ReLU) activation function. After the

ReLU activation, a max-pooling layer with a 2×2 kernel reduces the spatial size of the image, which also decreases the number of trainable parameters and the computational cost. Skip connections help the decoder produce better semantic features and improve gradient flow during backpropagation, enhancing the network's ability to learn representations. The bridge section connects the encoder and decoder, consisting of two convolutional layers with a 3×3 kernel and a subsequent ReLU activation function. The decoder takes the abstract representation from the encoder and bridge to generate the semantic segmentation mask. Each decoder block begins with a 2×2 transpose convolution, followed by the concatenation of the corresponding skip connection with the output of the transpose layer. After these two layers, a ReLU activation function is applied. The final decoder layer connects to a convolutional layer with a 1×1 kernel and uses a sigmoid or SoftMax activation function, depending on the number of classes being differentiated during segmentation.

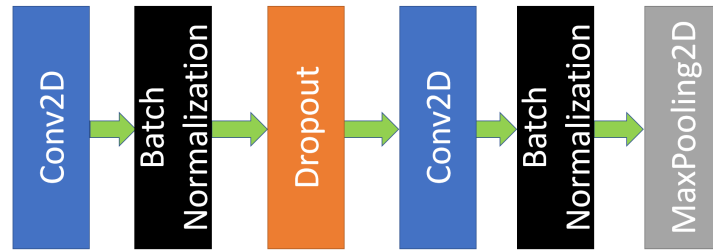


Figure 4.3: The structure of an encoder block, using Conv2D layers with kernel size 3×3 , ReLU activation function, and MaxPooling2D kernel size 2×2 .

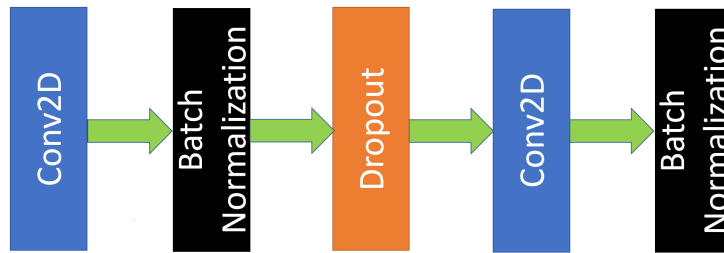


Figure 4.4: The structure of the network's bridge part, using Conv2D kernel size 3×3 , and ReLU activation function.

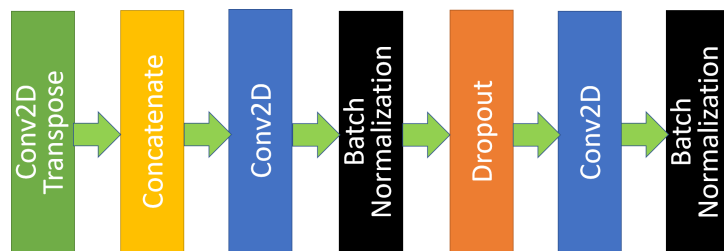


Figure 4.5: The structure of a decoder block, using Conv2DTranspose kernel size 2×2 , Conv2D kernel size 3×3 , and ReLU activation function.

My model employs four encoding blocks, as shown in Fig. 4.3. Each encoding block includes a convolutional layer with a 3×3 kernel, followed by a batch normalization layer and a dropout layer. The block continues with another convolutional layer with a 3×3 kernel, followed by a batch normalization layer and a max-pooling layer with a 2×2 kernel. The filter sizes for the four encoder blocks are 256, 128, 64, and 32, respectively.

The bridge component of my model, illustrated in Fig. 4.4, consists of a convolutional layer with a 3×3 kernel, followed by a batch normalization layer and a dropout layer. This is followed by another convolutional layer with a 3×3 kernel, and the final layer performs batch normalization. The filter size for the convolutional layers in the bridge is 16.

My network’s decoder consists of four blocks, as depicted in Fig. 4.5. Each block contains a transpose convolutional layer with a 2×2 kernel, a concatenate layer that merges the skip connection and the transpose layer’s output, a convolutional layer with a 3×3 kernel, a batch normalization layer, a dropout layer, another convolutional layer with a 3×3 kernel, followed by a final batch normalization layer to complete the block. The filter sizes for the four decoder blocks are 32, 64, 128, and 256, respectively. The transpose filter sizes also double from block to block, ranging from 16 to 128. The output layer is a convolutional layer with a 1×1 kernel and a filter size of 4, distinguishing the four classes in segmentation: background, white matter, gray matter, and cerebrospinal fluid. The network model processes six input images: three T1-weighted and three T2-weighted slices, incorporating neighboring slices in the segmentation of the current slice. The first and last slices are duplicated to provide neighboring slices for the first and last slices.

The network model uses sparse categorical cross-entropy as the loss function, optimized with the Adam optimizer algorithm [R195]. Training required no more than 100 epochs to achieve high-quality results.

4.3.6 First result

Table 4.1: Average values of the main accuracy indicators

Accuracy indicator	Tissue type		
	CSF	GM	WM
Dice score	0.9272	0.8867	0.8704
Sensitivity	0.9219	0.8917	0.8726
Specificity	0.9824	0.8950	0.9436
Precision	0.9351	0.8829	0.8700
Accuracy	All tissues		
	0.8908		

Table 4.1 shows the average accuracy metrics calculated for the ten individual MRI records. A global accuracy of over 0.89 suggests that nearly 11% of pixels are misclassified. Among the three tissue types, cerebrospinal fluid (CSF) has the highest overall segmentation accuracy. White matter (WM) shows the second-highest specificity, while gray matter (GM) has the best average scores for the other three metrics.

4.3.7 Second U-net network

One significant change since my previous work ([C8]) is the loss function. In this study, I implemented a custom loss function comprising two additive components. The first component is the categorical cross-entropy loss, while the second is a Dice score-based loss for the four classes (the three tissue types and the non-brain region). Each component is weighted equally, with a weight of 0.5 in the overall loss function. These adjustments were made by fine-tuning the function in TensorFlow.

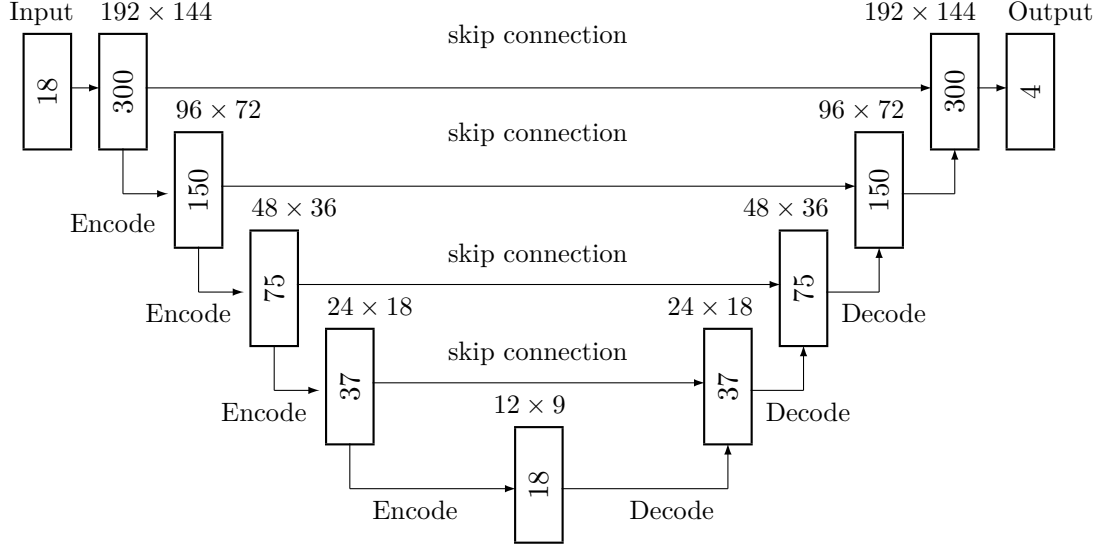


Figure 4.6: The proposed 2D U-net architecture.

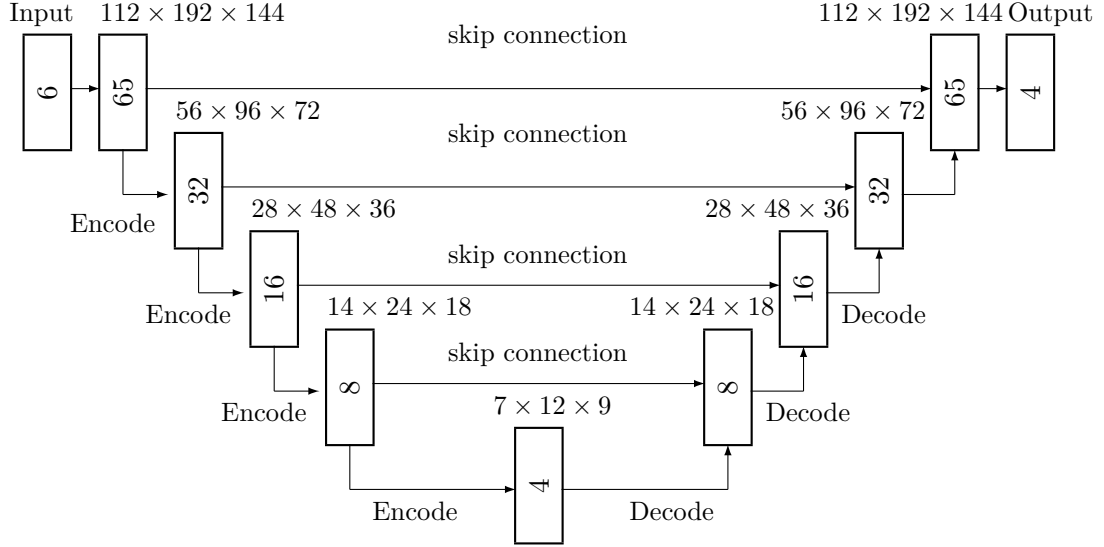


Figure 4.7: The proposed 3D U-net architecture.

The 2D U-net model, depicted in Fig. 4.6, consists of four encoder layers, a bridge layer, and four decoder layers. The input layer is a convolutional layer. When an arbitrary slice n is presented to the network, it also receives slices $n-1$ and $n+1$ from both the T1 and T2 data channels, totaling six channels. Additionally, these slices are concatenated into three examples, representing the red, green, and blue channels of color images, similar to typical CNN applications in image processing, despite working with single-channel grayscale images. Thus, the input shape becomes $192 \times 144 \times 18$. The output represents the segmentation for slice n , encoded in four channels for the four classes, resulting in an output shape of $192 \times 144 \times 4$. The encoder blocks have 300, 150, 75, and 37 channels, respectively. The bridge layer contains 18 channels, while the decoder blocks have 37, 75, 150, and 300 channels. The output layer has four channels and uses a SoftMax activation function. All other U-net layers use the exponential linear unit (ELU) activation function ([R196]). The structures of the encoder, bridge, and decoder blocks are shown in Fig. 4.8, with all convolutions operating in 2D. The encoder structure is as follows: a Conv2D layer with a 3×3 kernel, ELU activation, and same padding, followed by batch normalization, a dropout layer with a rate of 0.2, another Conv2D layer identical to the first, another batch normalization layer, and a MaxPooling layer with a pool size of 2×2 . The

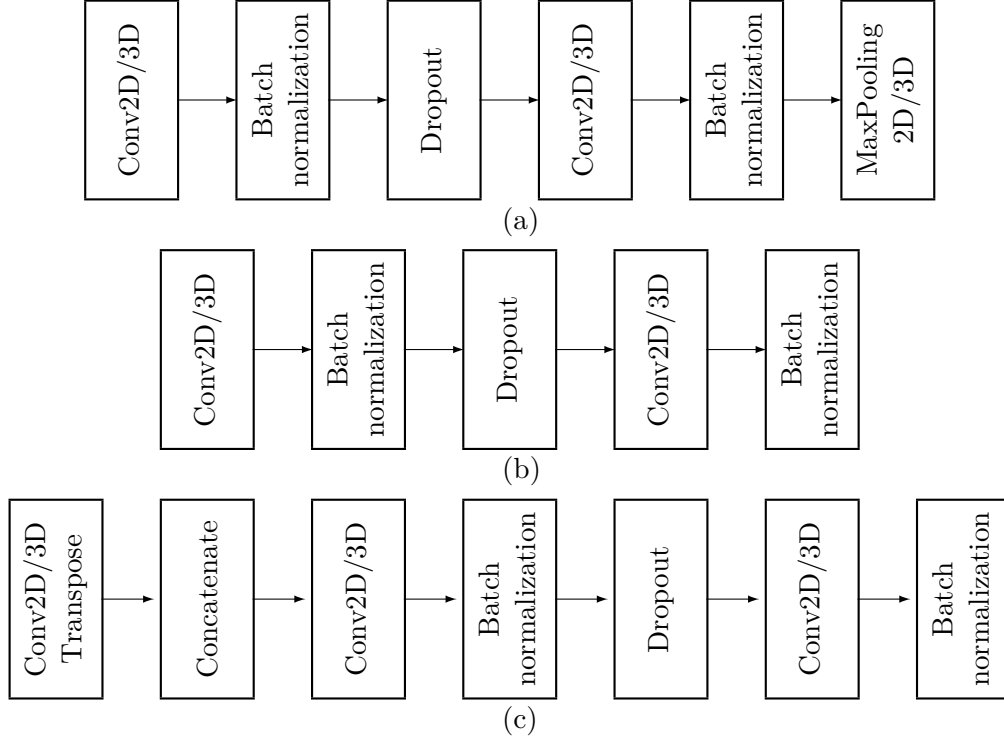


Figure 4.8: The structure of: (a) encoder block; (b) bridge part of network; (c) decoder block. Convolution works in 2D or 3D, depending on the architecture model.

bridge block has five layers: starting with the same Conv2D layer as the encoder, followed by batch normalization, a dropout layer, another Conv2D layer, and a final batch normalization layer. The decoder structure includes seven layers: first, a Conv2DTranspose layer with a 2×2 kernel and stride of 2×2 , a concatenate layer for connecting the skip connection, a Conv2D layer identical to those in other blocks, batch normalization, a dropout layer with a rate of 0.2, another Conv2D, and the final batch normalization layer in the block.

The 3D U-net architecture has the same number of encoder and decoder blocks as the 2D U-net, but with different configurations Figure 4.7. The input shape is $112 \times 192 \times 144 \times 6$ because it processes entire T1 and T2 volumes simultaneously, organized into three examples for each, corresponding to the red, green, and blue channels. The output shape is $112 \times 192 \times 144 \times 4$, representing the four-class segmentation of the entire volume.

The encoder blocks have 65, 32, 16, and 8 channels, respectively, while the bridge contains 4 channels. The decoder blocks are structured with 8, 16, 32, and 65 channels. As illustrated in Fig. 4.8, all convolutions are performed in 3D. The encoder block consists of six layers: starting with a Conv3D layer with a $3 \times 3 \times 3$ kernel and ELU activation, followed by batch normalization, a dropout layer with a 0.2 rate, another Conv3D layer identical to the first, additional batch normalization, and a MaxPooling 3D layer with a pool size of $2 \times 2 \times 2$. The bridge comprises five layers: a Conv3D layer identical to the encoder, batch normalization, a dropout layer with a 0.2 rate, another Conv3D layer like the first, and a final batch normalization layer. The decoder includes seven layers: beginning with a Conv3DTranspose layer with a $2 \times 2 \times 2$ kernel and stride, a concatenation layer for merging the skip connection, a Conv3D layer like the previous one, followed by batch normalization, a dropout layer with a 0.2 rate, another Conv3D layer, and concluding with a batch normalization layer.

4.3.8 Second results

All ten MRI records used in this study underwent preprocessing as outlined in Section 4.3.4. Subsequently, each record was tested individually using the “leave-one-out” method, where the

Table 4.2: Average accuracy indicator values obtained from the ten MRI records

Accuracy indicator	2D model			3D model		
	Tissue type			Tissue type		
	CSF	GM	WM	CSF	GM	WM
Dice score	0.939	0.905	0.890	0.950	0.915	0.898
Sensitivity	0.941	0.909	0.886	0.947	0.921	0.895
Specificity	0.983	0.912	0.955	0.988	0.919	0.958
Precision	0.937	0.902	0.895	0.954	0.910	0.902
Accuracy	All tissues			All tissues		
	0.908			0.918		

model was trained on the other nine records. For each segmentation outcome, the accuracy metrics were calculated and recorded for individual patients. Average metrics were then computed to provide a global measure of segmentation accuracy. These experiments were conducted separately using both the Conv2D and Conv3D network models.

Table 4.2 displays the average accuracy metrics obtained for the ten MRI records, separately for each network model. The global accuracies of 0.908 and 0.918 suggest a high-quality segmentation, with only 8-9% of pixels misclassified. This level of accuracy makes my algorithm competitive with the best solutions submitted to the iSeg-2017 challenge ([R174]). Among the three tissue types, CSF is most accurately segmented with a Dice score of 0.95, while WM has the lowest, yet still acceptable, Dice score of nearly 0.9. Specificity is the only metric where GM tissue has the lowest value. Comparing the two network architectures, the 3D convolution model consistently achieves higher scores, with an improvement of approximately 0.01 in all cases.

4.3.9 Spatiotemporal Convolutions: Unleashing Dynamics in Deep Learning

The advancement of deep learning has significantly transformed my ability to process and interpret complex data, with spatiotemporal convolutions exemplifying this change. These convolutions, an essential enhancement of traditional spatial convolutions, have become crucial for analyzing dynamic patterns and motion in video data, fundamentally changing video analysis and action recognition [R197].

Spatiotemporal convolutions surpass traditional spatial convolutions by extending beyond spatial dimensions to include the temporal dimension, which is vital for video data analysis. While spatial convolutions effectively capture local spatial relationships within individual image frames, they cannot account for the temporal dynamics essential to video sequences. Spatiotemporal convolutions address this limitation by integrating spatial and temporal information, enabling deep learning models to understand both appearances and movements.

This integration of spatial and temporal elements is accomplished by applying convolutions across three dimensions: two spatial and one temporal. Practically, each spatiotemporal convolutional kernel moves through the spatial dimensions and across the timeline of sequential frames. This enables the neural network to identify complex spatial patterns while capturing their temporal progression, effectively encoding movements, actions, and temporal dependencies essential for tasks like action recognition, video analysis, and gesture understanding.

The effectiveness of spatiotemporal convolutions is most apparent in action recognition, where accurately modeling motion sequences is critical. By utilizing spatiotemporal convolutions, deep neural networks can autonomously and adaptively learn intricate spatiotemporal features directly from raw video data, eliminating the need for labor-intensive, domain-specific handcrafted feature engineering.

Spatiotemporal convolutional network architectures typically consist of stacked convolutional layers, interspersed with pooling and nonlinear activation functions. A key feature is the use of 3D convolutional kernels, which span both spatial and temporal dimensions. This enables the network to capture not only static appearances but also the dynamic progression of actions and events over time.

In deep learning, spatiotemporal convolutions exemplify the powerful integration of neural networks with temporal dynamics. By allowing the simultaneous exploration of spatial and temporal dimensions, these convolutions have been transformative in various fields beyond action recognition, including video understanding, gesture analysis, and spatiotemporal anomaly detection.

In summary, the advent of spatiotemporal convolutions has expanded the scope of deep learning and ushered in an era of automated interpretation of complex temporal data. Their ability to integrate spatial patterns with temporal dynamics highlights the limitless possibilities that continue to emerge as I leverage the power of deep learning in a world rich with spatiotemporal data.

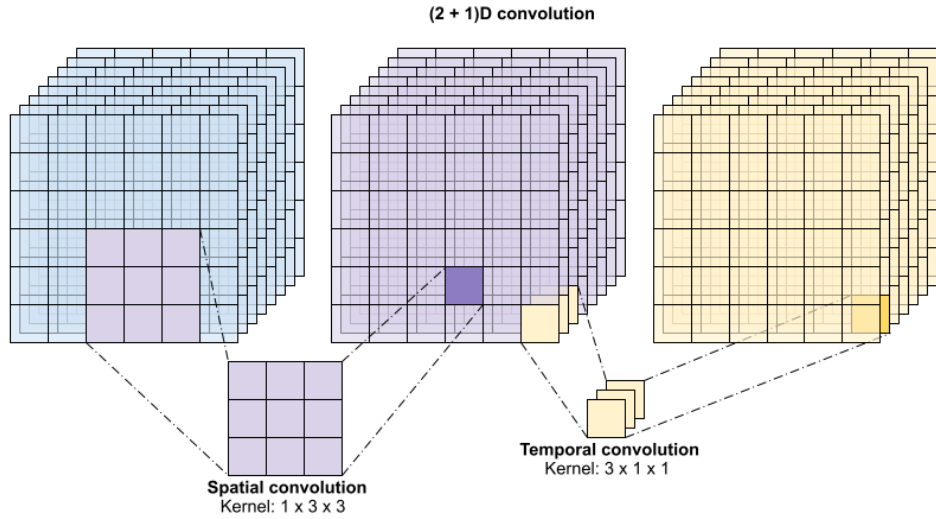


Figure 4.9: 2 plus 1 dimensional convolution

4.3.10 Third U-net

Overall

My entire U-net network is built on the experience I have accumulated [C8, J5] and incorporates the 2 plus 1 dimensional convolutional layer design originally proposed for video analysis [J7]. The network consists of four encoder and decoder sections, along with a bridge, skip connections, an input layer, and an output layer. On the input side, I merged the T1 and T2 channel images, resulting in six color channels by placing them side by side. Additionally, each volume contains 112 slices, with each image having dimensions of 192×144 pixels, giving me an input size of $112 \times 192 \times 144 \times 6$. As previously mentioned, the encoding section consists of four encoder blocks. The number of filters is halved at each step, as is the output image size. Consequently, the filter numbers are 65, 32, 16, and 8. The image sizes decrease progressively from $112 \times 192 \times 144$ to $56 \times 96 \times 72$, then to $28 \times 48 \times 36$, and finally to $14 \times 24 \times 18$. Each encoder block output has a skip connection to the corresponding decoder block at the same level, which also serves as an additional output for the encoder block.

Next is the bridge, which has an input size of $7 \times 12 \times 9$, representing the smallest compression in my network. The bridge block uses four filters, similar to the output layer. The resizing

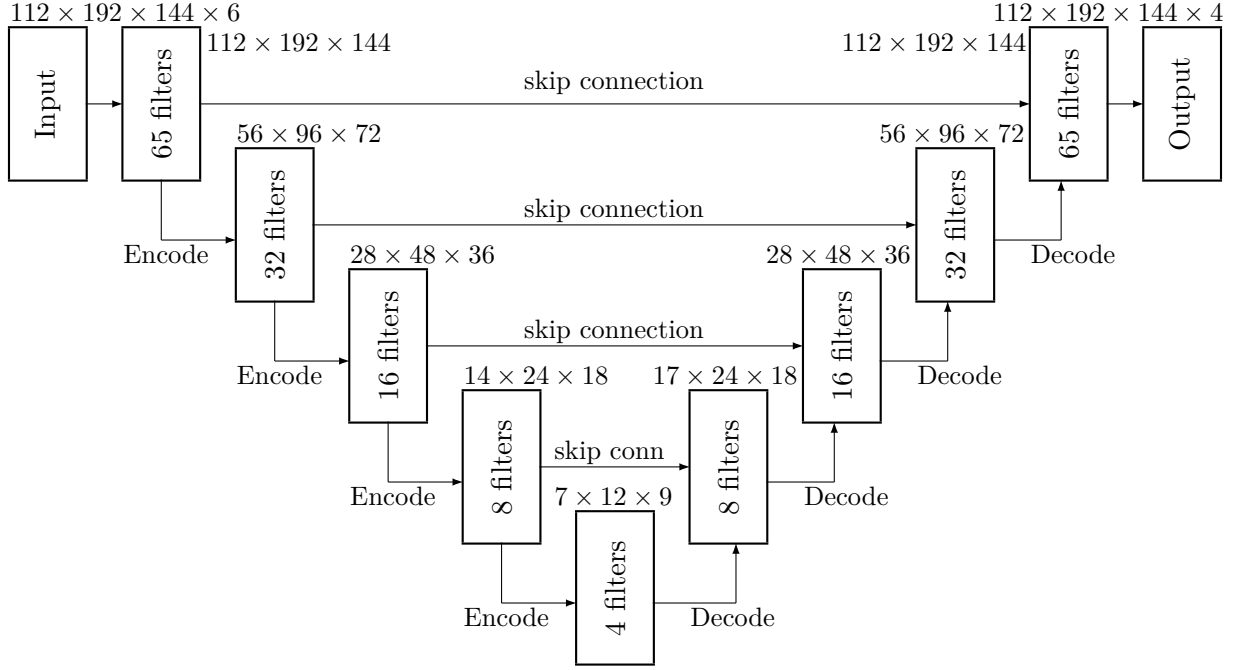


Figure 4.10: The proposed 3D U-net architecture.

process is then carried out by the decoder section. This section, like the encoder, contains four decoder blocks. However, as I move from the bottom up, the number of filters and image dimensions increase. The filter numbers are 8, 16, 32, and 65 in order, while the image sizes are $14 \times 24 \times 18$, $28 \times 48 \times 36$, $56 \times 96 \times 72$, and $112 \times 192 \times 144$, respectively. Each decoder block also utilizes the output from the encoder block at the same level, received via the skip connection.

Finally, the last layer is the classifier layer with four filters, corresponding to the four output classes (three tissue types and the surrounding space). Thus, my final output has a size of $112 \times 192 \times 144 \times 4$.

Encoder blocks

Figure 4.11 illustrates the structure of the encoder blocks. Each block contains four convolutional layers, two batch normalization layers, a dropout layer, and a max-pooling layer. Data entering the block first goes through a 3D convolutional layer with a unique kernel. Following the 2 plus 1D rule, the first convolutional layer processes data spatially with a kernel of size $1 \times 3 \times 3$. The output then moves to the next 3D convolutional layer, designed for spatial feature detection, with a kernel size of $3 \times 1 \times 1$, in line with (2+1)D convolution principles. The activation function used is ELU [R198], as it proved most effective in my previous studies.

A batch normalization layer follows, normalizing the data, then a dropout layer with a rate of 0.2 to prevent overfitting. Two more convolutional layers follow, structured similarly to the first two: the first uses a $1 \times 3 \times 3$ kernel, and the second uses a $3 \times 1 \times 1$ kernel with ELU activation. The number of filters in the convolutional layers is consistent throughout the block, as shown in Figure 4.10. Another batch normalization layer follows to further normalize the data. The output of this layer is sent to the skip connection branch, which connects to the corresponding decoder block. The final layer is a max-pooling layer that reduces the dimensions by half, with a pooling size of $2 \times 2 \times 2$. Finally, the output is passed either to the next encoder block or to the bridge.

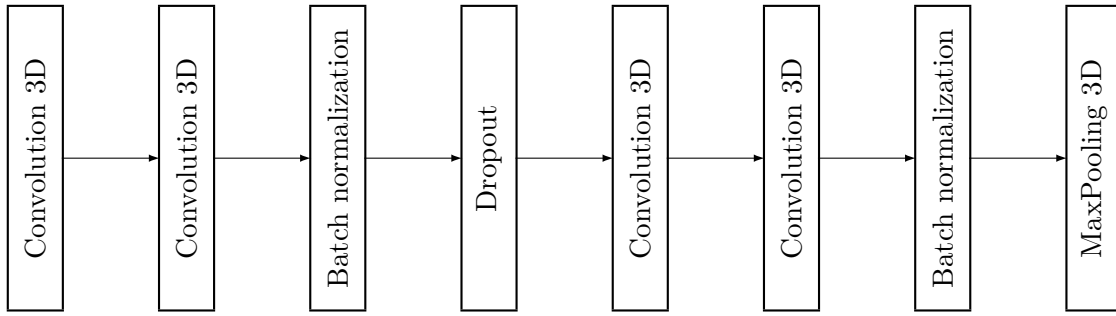


Figure 4.11: Structure of an encoder block.

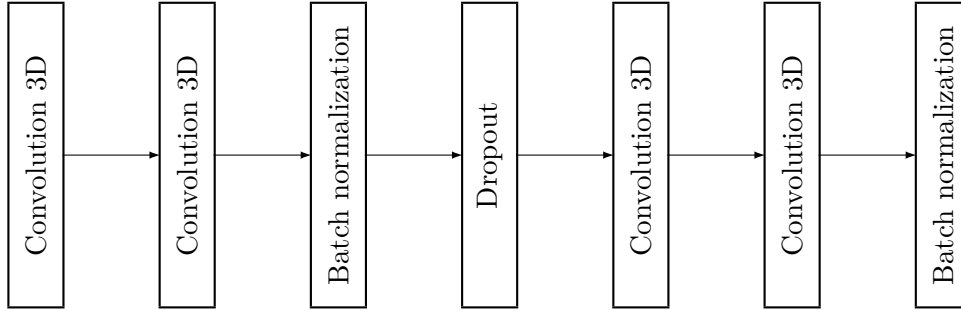


Figure 4.12: Structure of the bridge part.

Bridge

The next component is the bridge block Figure 4.12, which connects the encoder to the decoder and represents the smallest size within the network. In U-net, which functions as a type of autoencoder, this block is often referred to as the "key," reflecting the smallest representation typical in autoencoder networks. Its structure is similar to that of the encoder and decoder blocks, with only slight differences.

The bridge block consists of four convolutional layers, two batch normalization layers, and a dropout layer. When data enters this block, it first passes through a 3D convolutional layer with a $1 \times 3 \times 3$ kernel, similar to the first convolutional layer in the encoder. This is followed by a batch normalization layer to standardize the data. To prevent overfitting, a dropout layer with a rate of 0.2 is used next. Subsequently, two additional 3D convolutional layers are applied: the first with a $1 \times 3 \times 3$ kernel and the next with a $3 \times 1 \times 1$ kernel. A final batch normalization layer follows to normalize the data further. All convolutional layers employ the ELU activation function, and the number of filters is consistent across layers, as indicated in Figure 4.10.

Decoder

Finally, let's look at the structure of the decoder blocks Figure 4.13. These blocks contain most of the layers because they use the output from the encoder block at the same level and the output from either the bridge or a decoder block one level below. Each decoder block consists of four convolutional layers, two batch normalization layers, a Conv3DTranspose layer, a concatenate layer, and a dropout layer.

Here's a step-by-step breakdown of how the layers work and their functions. When data enters a decoder block, it first passes through the Conv3DTranspose layer. This data comes from a lower layer, either the bridge or another decoder block. The Conv3DTranspose layer increases the dimension of the deconvolution layer. Since the size of the first layer is half the size of the encoder block output at that level, I need to double the size, which is achieved by this layer. Thus, the kernel of this layer is $2 \times 2 \times 2$, similar to the encoder pooling layer.

Next, the concatenate layer combines the output of the previous layer with the skip connection value from that level, which comes from the encoder block before dimension reduction. This

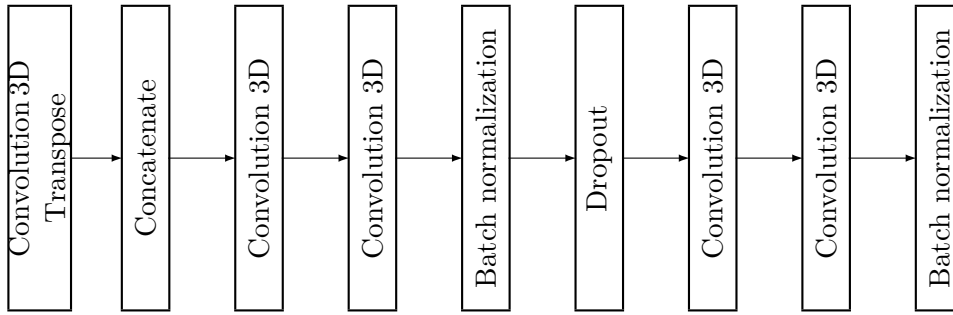


Figure 4.13: Structure of a decoder block.

is followed by two convolutional layers. The first convolutional layer searches for spatial features and uses a $1 \times 3 \times 3$ kernel. The second convolutional layer searches for temporal features and has a $3 \times 1 \times 1$ kernel.

A batch normalization layer follows to normalize the data, then a dropout layer to prevent overfitting. Next, there are two more convolutional layers, similar to the previous ones: the first uses a $1 \times 3 \times 3$ kernel for spatial feature detection, and the second uses a $3 \times 1 \times 1$ kernel for temporal feature detection. Finally, another batch normalization layer normalizes the data. The number of filters in the convolutional layers is consistent within a block, and all use the ELU activation function.

Classification layer

The classification layer has 4 filters. Although my task was to segment the 3 tissue types by classifying them at the pixel level, my data also includes a background class. This results in four total classes. This layer is a 3D convolutional layer with a kernel size of $1 \times 1 \times 1$, as it examines one pixel at a time. The SoftMax activation function, commonly used in classification tasks, is applied here. The network's final output, produced by this layer, has dimensions $112 \times 192 \times 144 \times 4$. This means that for each pixel, I obtain the predicted probabilities of the pixel belonging to each of the four classes. The final segmentation results are determined by selecting the class with the highest probability for each pixel.

Training and testing

The training and testing processes closely followed the methodology employed in my previous studies. Specifically, data from nine patients were used to form the training dataset, while the remaining one patient's data served as the testing dataset. This approach was cross-validated by repeating the process for each patient, resulting in a total of 10 test evaluations.

For training, I utilized my previously proposed cost function [J5], which combines sparse categorical cross-entropy and the Dice score cost with equal weighting. The Dice score cost was computed by subtracting the average Dice score of the four classes from 1. Optimization was carried out using the ADAM optimizer [R199], and the network was trained for 1000 epochs in each iteration.

During training, the model with the lowest cost function value was saved as the best model. For prediction, this saved model was reloaded, and its performance was evaluated using the test dataset.

4.3.11 Third results

Let us examine the results achieved during testing. Table 4.3 presents the ten patients alongside their respective F1 scores, Precision, and Recall metrics, arranged in ascending order of F1 scores.

Table 4.3: Recall, precision and F1 score for all patients, displayed in increasing order of the F1 score

Patient	Precision	Recall	F1 score
4	0.910571	0.906693	0.907219
7	0.915163	0.914503	0.914542
2	0.917713	0.917772	0.917736
5	0.919483	0.918829	0.918891
10	0.922817	0.921393	0.921721
9	0.923732	0.922788	0.922694
3	0.927507	0.927406	0.927385
6	0.928836	0.928748	0.928642
1	0.929377	0.929290	0.929122
8	0.932538	0.932624	0.932568

Table 4.4: Accuracy benchmark or rate of correct decisions, obtained for different patients, sorted in increasing order

Patient	Accuracy
4	0.906731
7	0.915281
2	0.917950
5	0.918972
10	0.921573
9	0.922976
3	0.927496
6	0.928893
1	0.929426
8	0.932671
mean	0.9225

The lowest F1 score was observed for patient 4, marking the only instance below 91%. This indicates that even in the worst case, my method achieves at least 90% accuracy in classifying different brain tissues. For this patient, Precision and Recall were also at their lowest values.

In contrast, the best results were recorded for patient 8, where the F1 score was 93.2%, representing a 2.5% improvement over the lowest score. Both Precision and Recall for this case also reached 93.2%. This is a notable enhancement compared to my previous work, where I were unable to surpass 93%.

Additionally, it is worth noting that only four patients had F1 scores below 92%, compared to six in my earlier tests, highlighting a clear improvement in performance. Table 4.4 displays the accuracy benchmarks achieved for various patients, arranged in ascending order. The bottom row of the table shows the mean accuracy rate, calculated as the average of the accuracy rates across all ten patients, which is 92.25%. Individual patient accuracy rates range from 90.6% to 93.2%, with the accuracy ranking aligning perfectly with the F1 score ranking.

Unlike earlier solutions, which consistently reported at least one patient with an accuracy rate below 90%, the proposed method achieves a minimum accuracy of 90.6%. The highest accuracy, 93.2%, was obtained for patient 8, the same patient who performed best in other benchmarks. The overall mean accuracy, 92.25%, is nearly 0.5% higher than the results reported in my previous work [J5].

Figure 4.14 illustrates the boxplot of the three benchmark metrics. The median values for all three metrics lie between 92% and 92.5%, consistent with the values presented in Table 4.3.

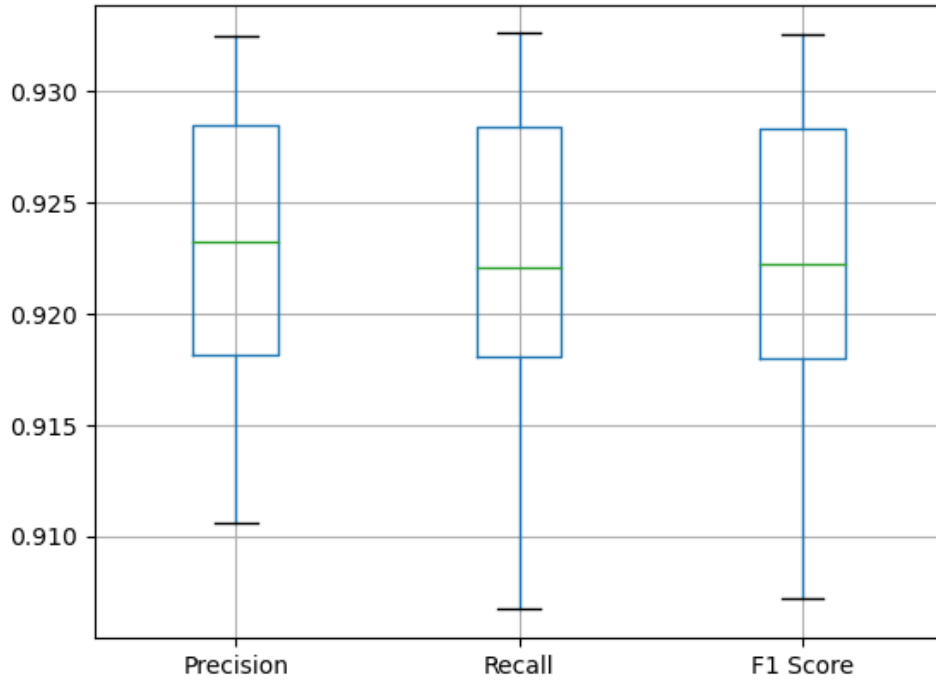


Figure 4.14: Boxplot of different segmentation benchmark metrics

Table 4.5: The distribution of various metrics values

	Precision	Recall	F1 Score
Avg	0.922774	0.922005	0.922052
Std	0.007010	0.007889	0.007732
Min	0.910571	0.906693	0.907219
Q1	0.918155	0.918036	0.918025
Q2	0.923274	0.922090	0.922207
Q3	0.928504	0.928412	0.928328
Max	0.932538	0.932624	0.932568

While the medians vary slightly, the interquartile range (IQR) for all three metrics remains similar. The lower quartile is positioned at 91.8%, and the upper quartile at 92.8%.

The lower endpoint for Recall and F1 score extends below 91%, whereas the Precision metric does not fall below this threshold. Conversely, the upper endpoint for all three metrics exceeds 93%, highlighting the strong performance across these benchmarks.

Let us analyze the metric values summarized in Table 4.5. Starting with the median values, Precision is 92.3%, while Recall and F1 score are both 92.2%. This minimal difference of 0.1% indicates that the test results exhibit very little variation. This observation is further supported by the standard deviation (Std) row in the table, which shows values of approximately 0.7%. Such low variability across the test cases confirms the absence of outliers and demonstrates the model's consistency in delivering high-quality results with minimal differences.

The mean values reinforce this consistency, with only the Precision column showing a noticeable deviation from the median. The Q1 (lower quartile) values, as reflected in the boxplot in Figure 4.14, are consistent across all three metrics at 91.8%. This marks a significant improvement over previous solutions, where this value represented the average performance. Similarly, the Q3 (upper quartile) values reveal minimal variation between metrics.

As highlighted in the boxplot analysis, the best test case achieved scores exceeding 93% across benchmarks. Even in the worst case, while Recall and F1 score dipped below 91%,

Table 4.6: Statistics of the ten confusion matrices obtained for individual patients

		Predicted		
		CSF	GM	WM
Actual	CSF	Min: 142971	Min: 3326	Min: 289
		Avg: 166034	Avg: 9018	Avg: 476
		Sdv: 15389	Sdv: 4580	Sdv: 198
		Max: 189750	Max: 20286	Max: 1029
		Q1: 151634	Q1: 6412	Q1: 379
		Q2: 168979	Q2: 7694	Q2: 455
		Q3: 176538	Q3: 9271	Q3: 475
	GM	Min: 1204	Min: 277932	Min: 15867
		Avg: 6199	Avg: 350375	Avg: 21216
		Sdv: 2893	Sdv: 31367	Sdv: 3691
		Max: 13262	Max: 402310	Max: 28401
		Q1: 4769	Q1: 338884	Q1: 18309
		Q2: 6045	Q2: 348180	Q2: 21157
		Q3: 6979	Q3: 365025	Q3: 23837
	WM	Min: 326	Min: 15842	Min: 162766
		Avg: 624	Avg: 24417	Avg: 220792
		Sdv: 221	Sdv: 5155	Sdv: 32762
		Max: 996	Max: 34771	Max: 272870
		Q1: 462	Q1: 22572	Q1: 207567
		Q2: 625	Q2: 23652	Q2: 219777
		Q3: 723	Q3: 26832	Q3: 239483

they remained well above 90%. This consistency across metrics underscores the robustness and reliability of the model's performance.

Figure 4.15 presents the benchmark values for individual records, plotted in ascending order for clarity. The first three panels display the Recall, Precision, and F1 score values for the three primary tissue types separately, while the fourth panel shows the overall accuracy for each record. The graphs highlight that CSF tissues are identified with the highest accuracy, as expected based on their intensity distributions. Recall values are noticeably higher for GM compared to WM tissues, while Precision is approximately equal for both. Overall accuracy, irrespective of tissue type, ranges from 90.7% to 93.3%, with minimal variance around the average of 92.25%.

Table 4.6 provides a summary of the confusion matrices for individual records. It is evident that the fewest misclassifications occur for CSF tissues, which aligns with expectations given the distinct intensity distributions of these tissues in T1 and T2 volumes. Misclassifications between CSF and WM are rare, while confusion between CSF and GM is more common. The most frequent misclassifications, however, occur between GM and WM tissues.

Interestingly, approximately the same number of GM pixels are misclassified as WM as WM pixels are misclassified as GM. However, since there are nearly twice as many GM pixels as WM pixels in the records, the accuracy of GM pixel identification is higher. This difference explains the observed disparity in F1 scores and Recall values between GM and WM tissues, as depicted in Figure 4.15.

Figure 4.16 presents the segmentation outcome of one of the ten records. The three green shades from darker to lighter ones represent the correctly identified pixels of CSF, GM, and WM tissues, respectively. Red color stands for misclassifications, regardless to the tissue types. It is easy to notice that most red pixels are situated at the boundary between two tissue types, where it is not so much obvious that the initial annotation was correct.

Table 4.7 summarizes a performance comparison between my previous work and a selection

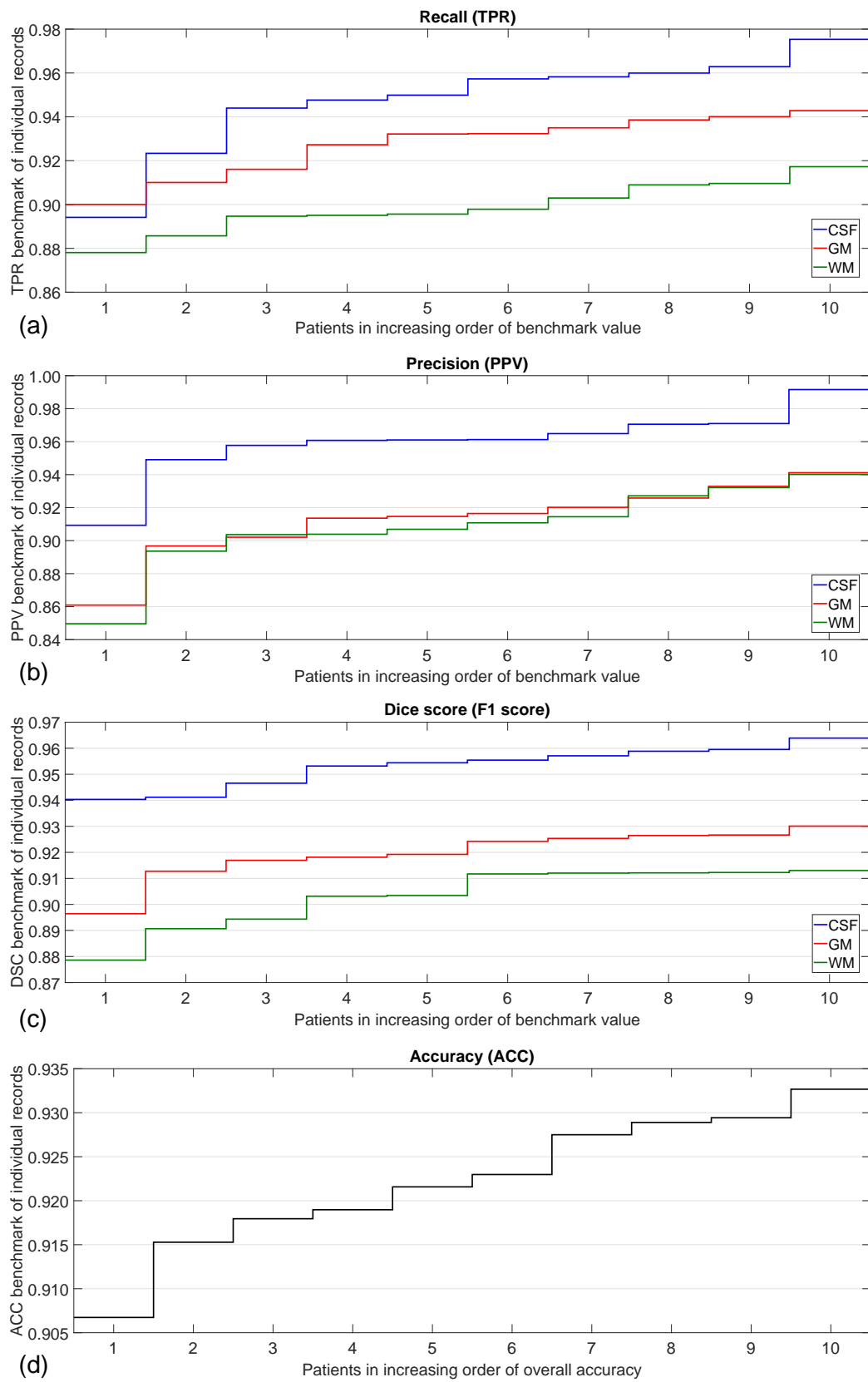


Figure 4.15: Benchmark values obtained for individual records and tissue types in panels (a) to (c); accuracy rates obtained for individual records in panel (d).

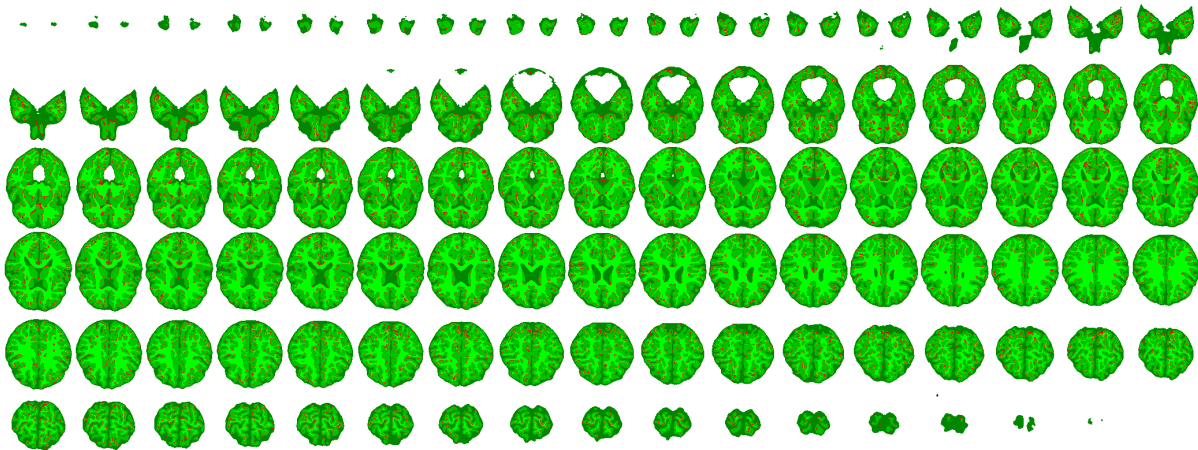


Figure 4.16: All slices of a segmented brain. The three shades of green from dark to light represent the correctly segmented pixels of the three main tissue types: CSF, GM and WM, respectively, while red color indicates misclassified pixels.

Table 4.7: Benchmark comparison with previous works and state-of-the-art methods

Paper	Method	F1-score				Accuracy
		CSF	GM	WM	Average	
Surányi et al. [R200]	random forest	0.879	0.835	0.796	0.837	0.833
Dénes-Fazakas et al [C8]	U-net 2D	0.927	0.886	0.870	0.894	0.890
Dénes-Fazakas et al [J5]	U-net 3D	0.950	0.915	0.898	0.921	0.918
Dénes-Fazakas et al [J6]	U-net (2+1)D	0.952	0.920	0.903	0.925	0.922
Qamar et al. [R189]	U-net 3D	0.957	0.920	0.905	0.927	N/A
Dolz et al. [R187]	CNN ensembles	0.957	0.918	0.897	0.924	N/A
Nguyen et al. [R201]	3D capsules U-net	0.949	0.911	0.902	0.920	N/A
Tran et al. [R202]	self-supervised 3D	0.949	0.914	0.907	0.923	N/A

of state-of-the-art methods for infant brain segmentation. The improvements over my earlier results are both clear and substantial, with all benchmark metrics showing higher values than in prior studies. Furthermore, the average F1-score column demonstrates that the proposed method is competitive with current state-of-the-art approaches.

4.3.12 Conclusion

In this work, I introduced a modified U-net-based approach for segmenting infant brain tissues from multi-spectral MRI data. The key innovation compared to my previous work was the replacement of 3D convolutions with (2+1)D convolutions in all encoder and decoder blocks of the U-net. This modification resulted in enhanced segmentation quality, outperforming my earlier methods and achieving competitive results across all evaluated statistical benchmarks. Future improvements could involve incorporating attention blocks into the model to further enhance the separation of GM and WM tissues, where most misclassifications currently occur.

4.4 Tumor classification of MRI scans

This chapter relates on another branch of MRI image processing research that I have been involved in, concerning the classification of brain tumors encountered in 2D MRI scans of adult brain.

4.4.1 Goal

The main goal of this chapter is to provide AIU-based solutions for the brain tumor classification problem. Given a set of MRI scans and the ground truth associated to them, an efficient and accurate method is needed to distinguish various types of the tumors.

4.4.2 Introduction

Each year, hundreds of thousands of individuals are diagnosed with brain tumors, with approximately 18,000 fatal cases reported annually in the United States alone [R203]. Early detection greatly enhances the chances of survival; however, the specific type of tumor also plays a critical role [R173]. Effective early detection does not necessarily require exact tumor segmentation; rather, it is essential that the detection process is sufficiently sensitive to identify even the smallest lesions. Brain tumors vary widely in size, shape, position, and visual characteristics, making detection difficult due to the distortions they cause in normal brain structures and the interference from various types of noise [R172].

In the past decade, there has been significant progress in brain tumor segmentation methods, driven largely by the Brain Tumor Segmentation (BraTS) Challenge, an annual event started in 2012 [R204, R205]. Initially, the BraTS challenge prompted researchers to explore a broad range of classical machine learning techniques [R206]. More recently, the field has undergone a major transformation with the advent of deep learning and various convolutional neural networks (CNNs), which have become the leading approaches in medical image analysis [R207, R208]. This shift highlights the substantial advancements and ongoing evolution in brain tumor segmentation methods.

While BraTS has primarily concentrated on gliomas, it is important to acknowledge the presence of various brain tumor lesions that require accurate classification for a well-rounded diagnostic approach. As a result, the problem of brain tumor segmentation has evolved as an extension of the BraTS challenges and has gained substantial traction in recent years. Numerous advanced solutions, mainly based on convolutional neural networks (CNNs), have emerged, achieving impressive results with accuracies often reaching 97-98% in the complex three-class brain tumor classification problem.

These cutting-edge methods not only utilize deep learning and convolutional networks but also integrate a range of sophisticated techniques:

1. **Generative Adversarial Networks (GANs):** Neelima et al. [R209] introduced GANs into the domain of brain tumor segmentation.
2. **Deep Hybrid Representation Learning:** Kanchanamala et al. [R210] proposed a deep hybrid representation learning approach, further enhancing the accuracy of brain tumor classification.
3. **Hybrid Deep Learning with Gabor Wavelets:** Rajeev et al. [R211] combined deep learning with Gabor wavelets to address brain tumor segmentation challenges.
4. **Adaptive Attention Mechanisms:** Mishra et al. [R212] and Reddy et al. [R213] introduced adaptive attention mechanisms to improve the segmentation of brain tumors.
5. **Deep Residual Learning Framework:** Mehnatkesh et al. [R214] proposed a deep residual learning framework for robust brain tumor segmentation.
6. **Recurrent CNNs:** Vankdothu et al. [R215] explored recurrent CNNs to address the intricacies of brain tumor segmentation.
7. **Parallel Deep CNNs:** Rahman et al. [R216] introduced parallel deep CNNs as a promising approach to enhance brain tumor segmentation.

Table 4.8: The structure of the image data set involved in the study

Tumor type	Patient count	Section plane			Total images
		Coronal	Sagittal	Transversal	
Glioma	89	437	495	494	1426
Meningioma	82	268	231	209	708
Pituitary	62	319	320	291	930
Total	233	1024	1046	994	3064

8. **Multi-Path Convolution and Multi-Head Attention Network:** Isunuri et al. [R217] devised a multi-path convolution and multi-head attention network for accurate brain tumor segmentation.

In this study, I explored a more sophisticated neural network for classifying different brain tumors. My approach involved small CNN and combining the U-net and CNN architectures.

4.4.3 Dataset

The dataset used to train and evaluate my CNN models in this study was initially compiled and publicly released by Cheng et al. [R218, R219]. It was meticulously gathered from 233 patients treated at two state-owned hospitals in Guangzhou and Tianjin, China, between 2005 and 2010. The dataset includes a total of 3064 T1-weighted contrast-enhanced brain MRI scans, each with a high resolution of 512×512 pixels and a pixel size of 0.49, mm in both dimensions.

The dataset is diverse, covering three types of brain tumors—Pituitary, Meningioma, and Glioma—and includes images from three different planes: axial, coronal, and sagittal. This variety provides a comprehensive view of the structural characteristics and localization of these tumors. The dataset’s distribution is as follows: 930 instances of Pituitary tumors, 708 instances of Meningioma tumors, and 1,426 instances of Glioma tumors, ensuring a well-balanced representation of these tumor types.

Each image is provided in MATLAB (.mat) format, which includes detailed information such as a description, a tumor mask outlining the tumor boundaries, a tumor class label specifying the tumor type, a tumor border for precise demarcation, and a unique patient ID for identification.

For a detailed overview of the dataset’s key attributes, see Table 4.8.

4.4.4 First Network architectures

In my experiments, I evaluated several pre-existing neural network architectures and developed a custom network for the task. I tested ResNet34, ResNet50, and VGG16 models. Given that most images in the dataset have a resolution of 512×512 pixels, this input size was initially used for all network architectures. Networks that demonstrated strong performance were also tested with reduced input sizes of 256×256 and 128×128 pixels.

For each architecture, I employed an 80/20 split of the images for training and evaluation, respectively, and conducted five tests per architecture to ensure every image served as evaluation data at least once. I saved two trained versions of each network to determine the best-performing model: one with the lowest cost function value and another with the highest accuracy on the training data. The cost function used was SparseCategoricalCrossentropy, and the optimizer was the Adam algorithm [R220].

The two network architectures proposed in this study are shown in Figure 4.17. Although their overall structures are quite similar, there are several key differences. One difference is the size of the input layer, which varies based on the dimensions of the images being processed.

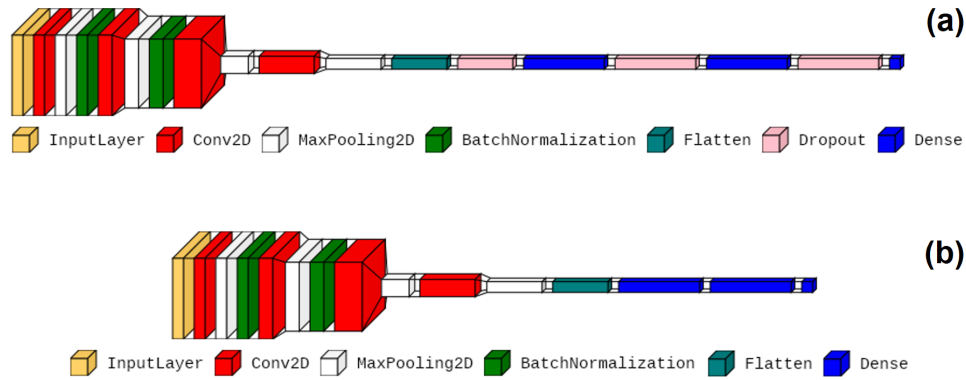


Figure 4.17: The proposed network architectures: (a) with dropout; (b) without dropout.

I explored various configurations for the number of Dense layers, trying out several different setups to optimize performance. Additionally, I experimented with kernel sizes, testing both 3 and 9.

My models were divided into two main configurations: one without a Dropout layer and one with a Dropout layer set to 0.2, positioned after the Flatten layer and before the Dense layers. Another variation occurred when processing 128×128 images; in this case, the final Maxpool layer was adjusted to 2×2 instead of the standard 4×4 . These modifications were made to assess their impact on the network's performance and to determine the most effective architecture for the given task.

The first model in my study features a total of 15 layers. It begins with an input layer that can handle image sizes of $512 \times 512 \times 3$, $256 \times 256 \times 3$, or $128 \times 128 \times 3$. This is followed by convolutional blocks, which are divided into two groups: those with a BatchNormalization layer and those without. The first two convolutional blocks include BatchNormalization, while the next two do not.

The initial convolution block consists of three layers. It starts with a Conv2D layer with 64 filters, a 3×3 kernel size, and ELU activation function with padding set to 'same'. I also tested a 9×9 kernel size in one of the configurations. This is followed by a MaxPool2D layer with a 4×4 size and a BatchNormalization layer.

The subsequent block mirrors the structure of the first but uses 128 filters and omits the BatchNormalization layer. The third block includes a Conv2D layer with 256 filters, maintaining all other parameters. This is followed by a MaxPool2D layer with a 4×4 size, and a similar block is repeated with 512 filters. For 128×128 images, the final MaxPool2D layer is adjusted to 2×2 .

Following the convolutional blocks, there is a Flatten layer that converts the data into a vector. This is followed by two Dense layers, with neuron counts varying between 32 and 4096 during testing, using ReLU activation. The final classification layer has 3 neurons with a SoftMax activation function.

Models with dropout layers differ by incorporating a dropout layer with a rate of 0.2 between the Flatten layer and the two Dense layers, resulting in a total of 18 layers for these configurations.

4.4.5 First results

The entire dataset of 3064 images was divided into five nearly equal groups, with each group being used as the test set in turn. During each test, four of these groups served as the training data, while the remaining group was used for evaluation. I investigated various network architectures as described. The performance of each network was assessed using the statistical indicators.

Table 4.9 provides a summary of the F1 scores obtained by each network across different test scenarios. In this table, \star VGG architecture refers to my custom implementation of the VGG network, while VGG and ResNet indicate the VGG16 and residual networks employed through transfer learning, respectively.

The third column of the table lists the input sizes for each network, representing the dimensions to which the original images were resized before being fed into the network. The fourth column details several parameters: [MA] denotes the maximum accuracy achieved during training; [mL] indicates the minimum loss recorded during training; [D] signifies whether a dropout layer with a rate of 0.2 was used during training; and [K9] indicates the use of 9×9 convolution kernels instead of the default 3×3 kernels.

F1 scores are reported as averages \pm standard deviations, reflecting the results from the five separate runs, each corresponding to one of the five image groups used as evaluation data. This detailed breakdown allows for a comprehensive comparison of network performances under various conditions.

Table 4.9: Average and standard deviation of the main accuracy indicator (F1 score) obtained by various network models. For each of the five runs, each performed using one group of input data for evaluation and the other four for training, the average of the three F1 scores (for classes Meningioma, Glioma, and Pituitary) were extracted. The average and the standard deviation of these five values are reported here for each network model.

CNN network	Dense neurons	Input size	Parameters	F1 score mean \pm stdev
\star VGG	256	256×256	[MA][D]	0.9827 ± 0.0066
\star VGG	2048	256×256	[MA][D][K9]	0.9814 ± 0.0042
\star VGG	2048	256×256	[MA][K9]	0.9814 ± 0.0029
\star VGG	1024	256×256	[MA][D]	0.9808 ± 0.0037
\star VGG	1024	128×128	[MA][D]	0.9795 ± 0.0018
\star VGG	256	256×256	[MA]	0.9789 ± 0.0054
\star VGG	4096	256×256	[MA][D]	0.9782 ± 0.0053
\star VGG	2048	128×128	[MA][D][K9]	0.9781 ± 0.0037
\star VGG	32	128×128	[MA]	0.9778 ± 0.0047
\star VGG	4096	128×128	[MA][D]	0.9776 ± 0.0046
\star VGG	2048	128×128	[MA]	0.9772 ± 0.0048
\star VGG	2048	128×128	[MA][D]	0.9772 ± 0.0025
\star VGG	1024	128×128	[MA]	0.9772 ± 0.0053
\star VGG	2048	128×128	[MA][K9]	0.9771 ± 0.0033
VGG	4096	128×128	[MA]	0.9769 ± 0.0066
ResNet	1000	512×512	[MA]	0.9767 ± 0.0062
\star VGG	2048	256×256	[MA][D]	0.9766 ± 0.0044
\star VGG	2048	256×256	[mL][K9]	0.9763 ± 0.0075
\star VGG	4096	256×256	[MA]	0.9762 ± 0.0026
\star VGG	4096	128×128	[MA]	0.9761 ± 0.0045
\star VGG	2048	256×256	[MA]	0.9760 ± 0.0055
\star VGG	256	128×128	[MA][D]	0.9759 ± 0.0047
\star VGG	2048	512×512	[MA]	0.9758 ± 0.0065
\star VGG	1024	512×512	[MA]	0.9758 ± 0.0035
\star VGG	32	256×256	[MA]	0.9753 ± 0.0074
\star VGG	1024	256×256	[MA]	0.9748 ± 0.0060
\star VGG	1024	256×256	[mL]	0.9742 ± 0.0031
\star VGG	256	128×128	[mL]	0.9742 ± 0.0103
\star VGG	4096	512×512	[MA]	0.9740 ± 0.0085

★VGG	256	128×128	[MA]	0.9739 ± 0.0057
★VGG	2048	512×512	[MA]	0.9733 ± 0.0056
★VGG	256	512×512	[MA]	0.9732 ± 0.0062
★VGG	2048	512×512	[mL][D][K9]	0.9718 ± 0.0093
★VGG	32	256×256	[mL]	0.9717 ± 0.0083
★VGG	256	256×256	[mL]	0.9714 ± 0.0067
★VGG	256	128×128	[mL]	0.9711 ± 0.0059
VGG	4096	256×256	[MA]	0.9711 ± 0.0041
ResNet	1000	512×512	[MA]	0.9703 ± 0.0038
★VGG	2048	128×128	[mL][K9]	0.9687 ± 0.0101
★VGG	2048	512×512	[mL][K9]	0.9681 ± 0.0071
★VGG	2048	128×128	[mL]	0.9680 ± 0.0080
VGG	4096	128×128	[mL]	0.9677 ± 0.0067
★VGG	256	512×512	[mL]	0.9677 ± 0.0117
VGG	4096	128×128	[MA]	0.9672 ± 0.0109
★VGG	2048	128×128	[mL][D]	0.9671 ± 0.0053
★VGG	4096	128×128	[mL][D]	0.9666 ± 0.0051
★VGG	256	256×256	[mL]	0.9665 ± 0.0080
★VGG	1024	256×256	[mL][D]	0.9664 ± 0.0085
★VGG	1024	512×512	[mL]	0.9660 ± 0.0110
★VGG	2048	256×256	[mL]	0.9660 ± 0.0142
★VGG	4096	256×256	[mL][D]	0.9658 ± 0.0115
★VGG	32	512×512	[MA]	0.9653 ± 0.0200
★VGG	2048	256×256	[mL][D]	0.9651 ± 0.0050
★VGG	4096	256×256	[mL]	0.9649 ± 0.0086
★VGG	1024	128×128	[mL]	0.9646 ± 0.0088
★VGG	4096	512×512	[mL]	0.9644 ± 0.0065
ResNet	1000	512×512	[mL]	0.9642 ± 0.0121
★VGG	32	128×128	[mL]	0.9639 ± 0.0109
VGG	4096	512×512	[MA]	0.9638 ± 0.0185
★VGG	1024	128×128	[mL][D]	0.9628 ± 0.0106

The results show that decreasing image size is beneficial, as it removes irrelevant features from the images. However, scaling the images to excessively small sizes is not advisable. None of the top 10 performing architectures utilized the original image size of 512×512 . Using 512×512 images initially was found to slow down training, as this size contains a lot of extraneous information.

It is evident from the table that only three models in the top 10 had an input image size of $128 \times 128 \times 3$, suggesting that excessively small images are not effective. The optimal resolution appears to be $256 \times 256 \times 3$. Furthermore, models retrieved based on maximum accuracy during training generally outperformed those selected based on the minimum cost function. Since the dataset is well-balanced, models emphasizing maximum accuracy are more suitable.

Another key observation from the table is that incorporating a dropout layer enhances training by reducing the likelihood of overfitting. Additionally, my findings suggest that pre-existing network architectures are not always the best solution for every problem. In my study, a simplified network structure similar to the VGG model yielded better results than more complex, pre-designed models. This may be attributed to the limited number of images available for training and the insufficient time allocated for training the pre-existing models, which could have benefited from more than 100 epochs.

In the initial comprehensive table, a summary of all the tested network models are presented, organized in descending order of their average F1 score. The data reveals that models utilizing

512 \times 512 image resolution generally perform worse compared to those with other resolutions. Specifically, models with a 256 \times 256 resolution outperform those with a 128 \times 128 resolution. This suggests that resizing and compressing images can improve network performance, although resizing to excessively small dimensions is not optimal. Keeping images at their original 512 \times 512 resolution proves to be problematic as they are too large and contain excessive, unnecessary features for effective model training.

Interestingly, commonly used model architectures did not achieve as favorable results as the custom architectures I employed. This is because the standard models are often designed for more complex images and have larger, deeper networks, which can hinder their ability to focus on specific tasks. The smaller networks I proposed, offering faster training and better suitability for this specific problem, outperformed the traditional models.

The table also indicates that very large or very small dense layers are less effective. Dense layers with 256, 1024, or 2048 neurons demonstrated optimal performance. Models with these configurations achieved average F1 scores exceeding 0.98 at a 256 \times 256 resolution. Furthermore, including a dropout layer generally enhanced model accuracy, as evidenced by its presence in four of the top five models. On average, models with dropout layers performed better than those without.

Regarding kernel size, the results are somewhat nuanced. The model with the highest average F1 score used a 3 \times 3 kernel. However, the second and third highest models employed 9 \times 9 kernels. These models showed lower variance compared to the top-performing model but had slightly lower average F1 scores. This suggests that while kernel size does not significantly affect average accuracy, it may help to surpass a certain minimum accuracy threshold.

4.4.6 L-net (U-net plus CNN), Second Architectures

In this study [J6], I proposed a complex network model, illustrated in Figure 4.18, based on the principle that neural networks can improve their learning when trained on multiple tasks concurrently. The architecture features a U-net and a CNN arranged in a unique cascade. The input MRI image is first processed by the U-net, which generates a predicted brain mask at its output. The output from the penultimate layer of the U-net (just before the final output layer) is then fed into the CNN, instead of the original image, allowing the CNN to predict the tumor type based on the features extracted by the U-net. The U-net is trained to perform segmentation using the provided tumor masks, while the CNN is trained to accurately classify the tumor type from the annotated images. Both parts of the architecture are trained simultaneously, with each training epoch adjusting the weights of both networks in response to each batch of images, ensuring they learn in parallel.

U-net in L-net

Our U-net, depicted in Figure 4.19, consists of four encoder blocks and four decoder blocks. These blocks are connected at the bottom by a bridge and include skip connections at each level. Unlike the original U-net, my model features an additional convolution block with nine filters, positioned right before the output layer, which contains two filters (marked in green) that produce the predicted segmentation output. The encoder blocks use 64, 32, 16, and 8 filters, respectively, while the decoder blocks mirror this in reversed order. The bridge uses the smallest number of filters, four, as it represents the smallest feature map within the U-net. Image sizes are not specified in Figure 4.19, since multiple resolutions were used during evaluation. Each encoder block reduces the image size by half in both dimensions, while each decoder block doubles it, restoring the original image size at the U-net's output. The CNN portion of the L-net connects after the green-marked layer and uses this as its input.

The structure of the encoder blocks is shown in Figure 4.20, with four such blocks forming the encoder branch of the U-net. Each block is composed of six layers: two Conv2D layers, two

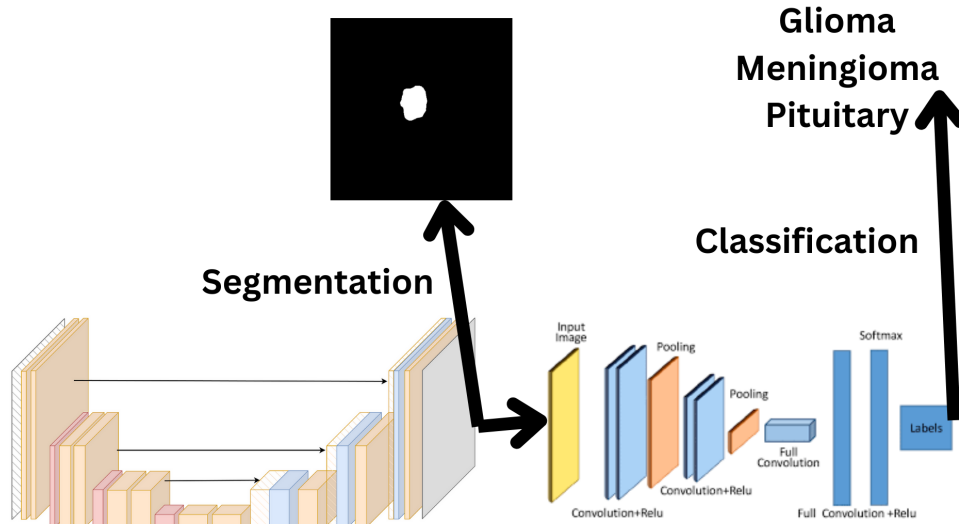


Figure 4.18: An overview of the L-net architecture, combining segmentation and classification tasks. The model first segments the tumor region using a U-net structure (**left**), where the black image represents the segmentation mask as the output. It then classifies the tumor, based on the features extracted by the U-net, into one of three categories: Glioma, Meningioma, or Pituitary (**right**), using a CNN.

batch normalization layers, a dropout layer, and a MaxPooling2D layer. The data flow within the block is detailed in Figure 4.20. First, the data passes through a Conv2D layer [R221], followed by batch normalization to standardize the output. Next, a dropout layer is applied to help prevent overfitting [R222], with a dropout rate of 0.2. The second Conv2D layer comes after this, followed by another batch normalization layer. Finally, the MaxPooling2D [R223] layer reduces the image dimensions. The MaxPooling2D layer uses a 2×2 matrix, halving the image size compared to the input. The skip connection retrieves the output from the second batch normalization layer, while the reduced-size output from the MaxPooling2D layer is passed to the corresponding decoder block. Both Conv2D layers utilize the ELU [R224] activation function. The number of filters in each encoder block is shown in Figure 4.19. The convolution kernels have a size of 3×3 , and their fill factor is set so that the data size remains unchanged within the convolution layers.

The bridge component of my U-net network, depicted in Figure 4.21, serves as the connection between the encoder and decoder branches. Its structure closely mirrors that of the encoder blocks, with the key difference being the absence of a MaxPooling2D layer at the end of the bridge. As a result, the image size is not further reduced at this stage. The blocks of the decoder branch in the U-net are structured as shown in Figure 4.22. These are the most complex blocks within the U-net, as they process the output from either the bridge or the previous decoder. The lower-level data is first upscaled by a Conv2DTranspose layer, then concatenated with the data from the corresponding skip connection at the same level, which is handled by the Concatenate layer.

Next, the data passes through two Conv2D layers, each followed by batch normalization, with a dropout layer in between, set to a dropout rate of 0.2. The Conv2DTranspose layer uses a 2×2 kernel matrix and a 2×2 stride, which doubles the size of the incoming data from the previous block. The number of filters in this layer matches the filter count of the Conv2D layers in the same block. Like the encoder blocks, the Conv2D layers use 3×3 kernels and the ELU activation function. The final decoder block produces output images that match the original input size of the U-net.

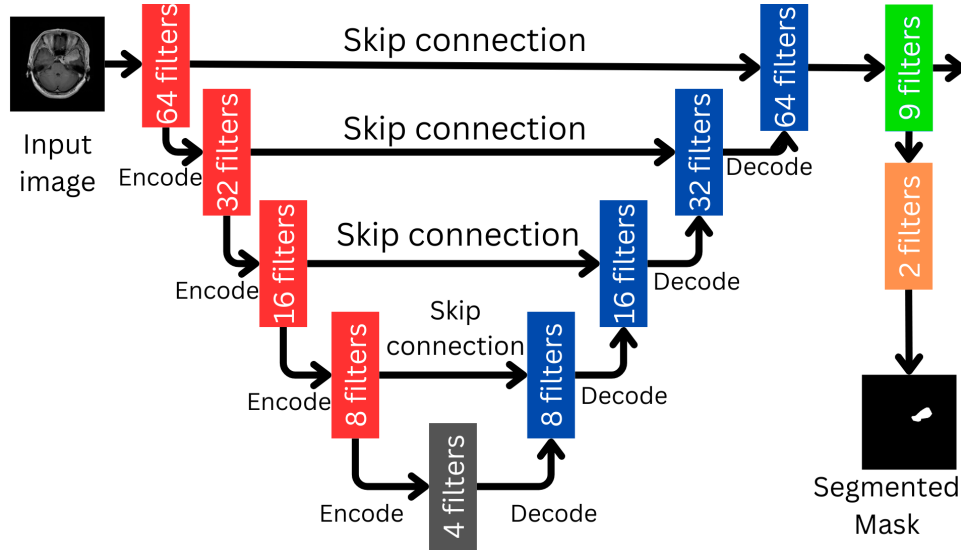


Figure 4.19: The U-net module, which forms a key part of the overall L-net architecture.

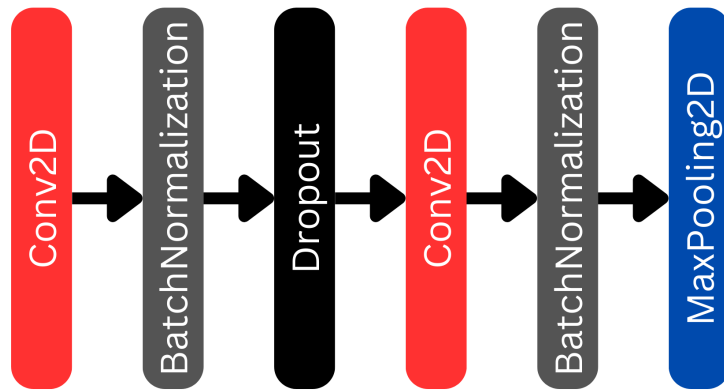


Figure 4.20: Encoder blocks of the U-net part of the L-net.

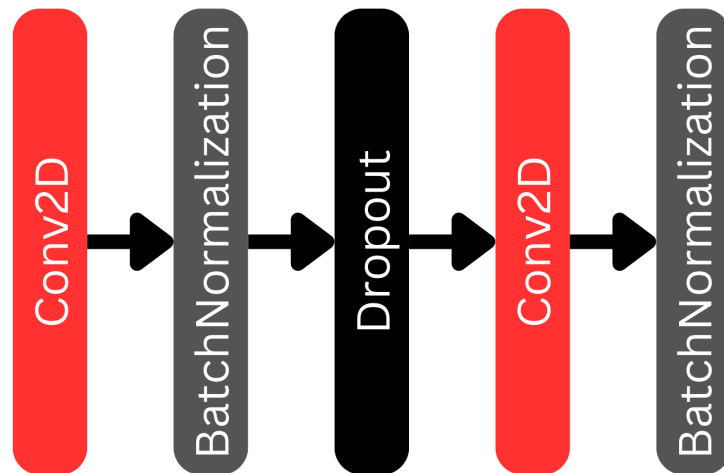


Figure 4.21: The bridge section within the U-net component of the L-net architecture.

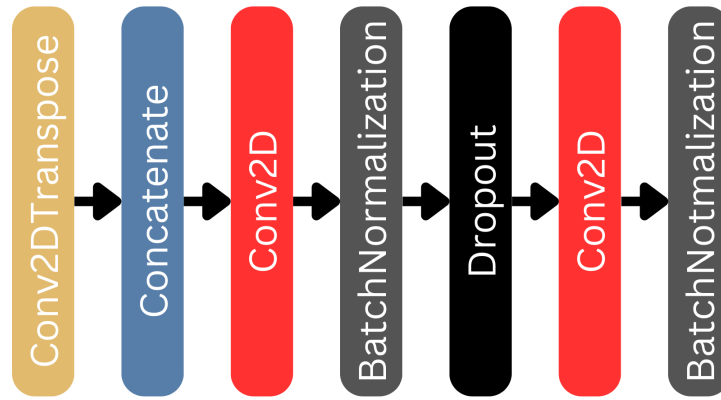


Figure 4.22: The decoder blocks within the U-net component of the L-net architecture.

CNN in L-net

The convolutional component of the L-net architecture is responsible for classification, as shown in Figure 4.23. As depicted in Figure 4.18, the CNN is connected directly after the U-net. Specifically, Figure 4.18 shows the connection point, where the output from the penultimate U-net layer, which consists of nine filters, is passed to the CNN's input. The CNN is divided into two parts: a convolutional section for feature extraction and a classifier section for performing the classification.

The structure of the CNN blocks is shown in Figure 4.23. The convolutional part contains four blocks with 64, 128, 256, and 512 filters, respectively. The classifier section begins with a flatten layer that converts the feature maps into a column vector, followed by three dense layers. The first two dense layers have 256 neurons each, while the final dense layer, which performs the classification, has three neurons corresponding to the three tumor classes to be distinguished.

These blocks, as mentioned earlier, come after the flatten layer, meaning they only operate on column vectors. The first layer in each block is a dropout layer with a rate of 0.2, added to reduce the risk of overfitting. Following that is a dense layer, which functions as a set of vector neurons. The number of neurons in each block matches the neuron count in the corresponding layer, as shown in Figure 4.23. The activation function for these blocks is ReLU [R225], except for the final layer, which uses the SoftMax activation function [R90] to predict probabilities for the classification task.

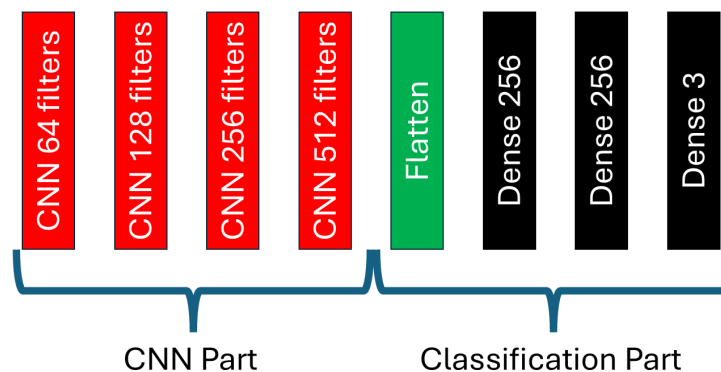


Figure 4.23: The detailed architecture of the CNN component that forms a crucial part of the overall L-net structure.

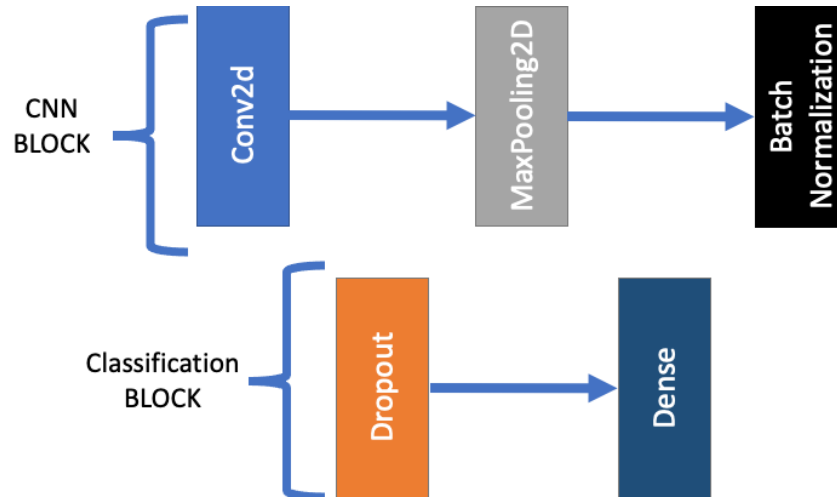


Figure 4.24: The individual blocks of the CNN component within the broader L-net architecture.

4.4.7 Training and testing

In my experiment, I evaluated various image resolutions to determine which one yields the best performance for my model. I aimed to identify the optimal resolution and assess whether I could surpass the results from my previous tests. Given the complexity of the network, I conducted training and testing using the following image resolutions: 16×16 , 32×32 , 64×64 , and 128×128 .

I did not evaluate the segmentation results in this study, as I worked with smaller image sizes compared to the original 512×512 resolution of the dataset. Therefore, segmentation results at reduced image sizes would be misleading. Although the model also produced segmentation outputs, my primary focus remained on classification.

I employed cross-validation in my analysis, similar to my previous tests. Specifically, 80% of the data was used for training, while 20% was reserved for testing. Cross-validation was performed by cycling through each tumor type, with every 5th item being allocated to the test set and the remaining items to the training set. The test set started with index 0 and cycled through to index 4. Consequently, I conducted 5 separate tests for each model configuration. Training was carried out over 1000 epochs using the Adam optimizer.

4.4.8 L-net Results

The entire set of 3064 images was used in a five-fold cross-validation process, with the same five nearly equal groups of images serving as the testing data in rotation while the model was trained on the remaining four groups. The proposed L-net model was evaluated under various hyperparameter settings, and its classification performance was compared to baseline models.

Segmentation results were not analyzed since the L-net operated on images smaller than the original 512×512 resolution of the dataset. Under these conditions, the segmentation results would not be reliable. Although the model produces a segmentation of the tumor in the input images, I focused solely on the classification performance, as in the previous test described in Section 4.4.5. The results are presented in the following tables and figures.

Table 4.10 presents the precision, recall, F1 score, and accuracy values achieved by the L-net model, along with the mean and standard deviation for each metric across the five folds of cross-validation. For the first three metrics, the results were averaged across the three different classes, and then the mean and standard deviation of these averages were reported in the table. For accuracy, a single overall value was calculated for each fold, representing the proportion of correct predictions.

The last row of the table also shows the best performance benchmarks obtained by the baseline model. It is evident that even at the lowest image resolution, the proposed L-net model

Table 4.10: Precision and recall values obtained by the L-net model.

Image Size	Precision Mean \pm Std	Recall Mean \pm Std	F1 Score Mean \pm Std	Accuracy Mean \pm Std
16×16	0.986743 ± 0.002653	0.986622 ± 0.002665	0.986633 ± 0.002656	0.986622 ± 0.002665
32×32	0.989924 ± 0.002099	0.989884 ± 0.002121	0.989888 ± 0.002106	0.989884 ± 0.002121
64×64	0.991222 ± 0.001860	0.991189 ± 0.001853	0.991187 ± 0.001865	0.991189 ± 0.001853
128×128	0.996761 ± 0.001155	0.996737 ± 0.001153	0.996738 ± 0.001154	0.996737 ± 0.001153
Baseline	0.9817 ± 0.0068	0.9838 ± 0.0063	0.9827 ± 0.0066	0.9827 ± 0.0066

outperforms the previous modified VGG model running with its optimal settings. As image resolution increases, the performance benchmarks improve significantly, with all values trending closer to 1. Additionally, the L-net model demonstrates greater stability compared to the VGG architecture, as indicated by the lower standard deviation of its accuracy metrics. At its peak performance, the L-net model results in four times fewer misclassifications than the VGG model.

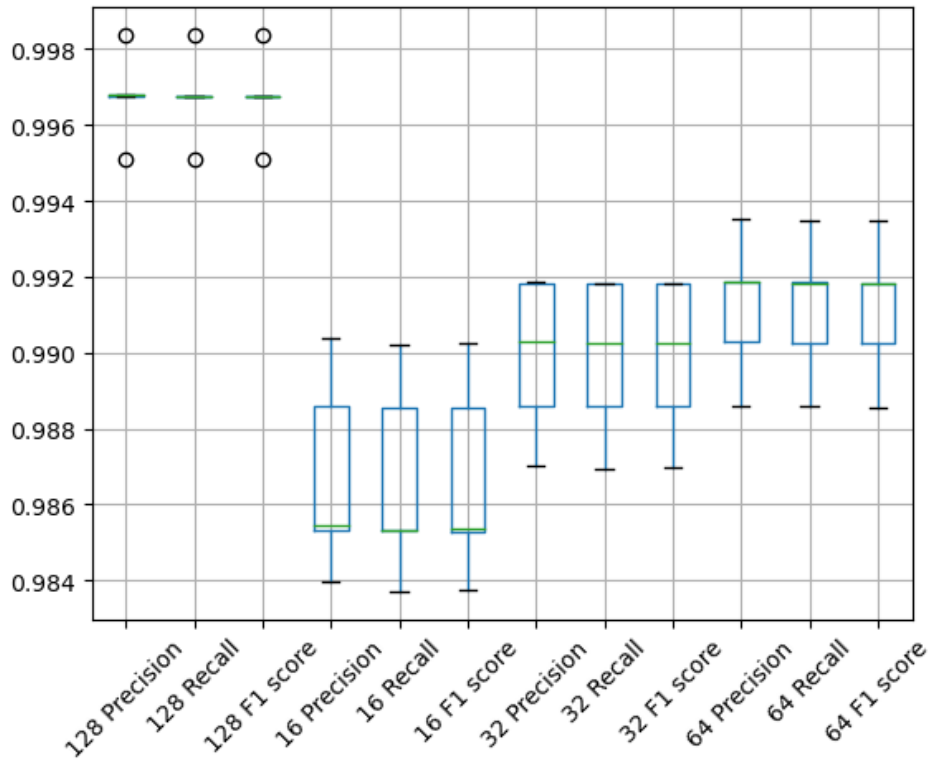


Figure 4.25: A boxplot for the precision, recall, and F1 score metrics. Lower axis indicates the image size and metric as well.

Figure 4.25 shows the boxplots of different metrics at various image resolutions. As the image resolution increases, all performance benchmarks make significant improvements, steadily approaching 1. A clear step-like increase can be seen: the median values rise, and the interquartile range (IQR) narrows.

At the lowest resolution, L-net slightly outperforms the best results of the previously modified VGG model. As the resolution doubles, the minimum values of the benchmarks become equal to or higher than the median obtained at the smaller resolution. With further increases in resolution, the range between quartiles continues to shrink, resulting in reduced variance across the five test cases. This indicates that the network's generalization ability becomes increasingly stable.

Table 4.11: AUC benchmarks obtained for different tumor classes

Image Size	AUC Meningioma Mean \pm Std	AUC Glioma Mean \pm Std	AUC Pituitary Mean \pm Std
16×16	0.996590 ± 0.001485	0.997482 ± 0.001324	1.000000 ± 0.000000
32×32	0.998341 ± 0.000495	0.997849 ± 0.000825	1.000000 ± 0.000000
64×64	0.997923 ± 0.001722	0.998192 ± 0.001595	0.999766 ± 0.000524
128×128	0.999646 ± 0.000229	0.999554 ± 0.000664	0.999992 ± 0.000017

Notably, there is a sharp and sudden decrease in the IQR at the 128 resolution, suggesting that at this point, the network achieves a high level of accuracy and consistency, significantly reducing variability in the test results. This leads to more reliable and precise outcomes. In other cases, there is already minimal variance in the tests. For example, at 64 resolution, the variance is between 0.990 and 0.992. At 128 resolution, three test cases cluster closely around 0.997, with two outliers slightly below 0.996 and above 0.998.

Table 4.11 shows the AUC values for each tumor class, along with the mean and standard deviation from the five cross-validation folds. Each row represents a specific image resolution, arranged in ascending order of image size. Across all resolutions, the Pituitary tumor class consistently achieves very high AUC values, often reaching 1 at lower resolutions. This indicates that the models learn this class most effectively, with minimal confusion between it and other classes, as reflected by its low variance.

For the Meningioma and Glioma classes, there is no significant difference in their mean AUC values, with either class occasionally having a higher value depending on the resolution. Notably, the AUC for the Glioma class surpasses the baseline model's benchmark only at resolutions of 64×64 and higher.

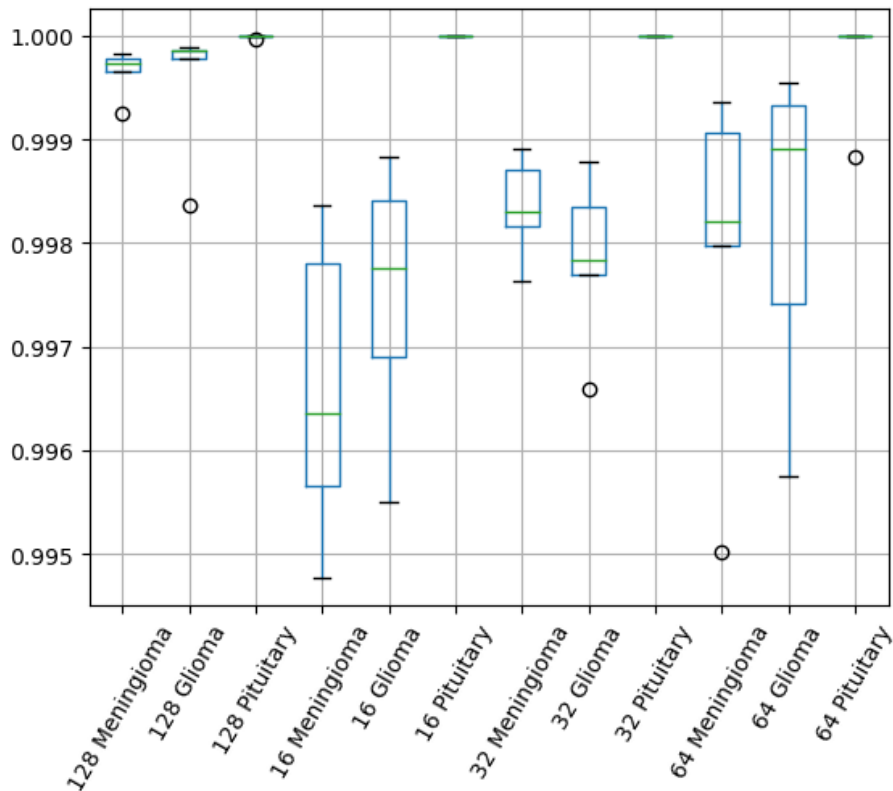


Figure 4.26: Boxplot for AUC metrics obtained for different tumor types and various imagesizes.

Figure 4.26 displays the boxplots of the AUC metrics outlined in Table 4.11. Across all test cases, the Pituitary class consistently hovers around an AUC of 1, indicating that it is easily distinguishable from the other classes. For the Glioma class, the lowest performance is observed at the 16×16 resolution, with the minimum AUC dropping to 0.9955. The median value is close to, but not quite reaching, 0.998, meaning that in half of the tests, the Glioma classification did not achieve an AUC of 0.998 at this resolution. Interestingly, at the 32×32 resolution, Glioma tumors were detected less accurately than Meningioma tumors, a trend not seen at other resolutions. At the 64×64 resolution, the median AUC value increases to nearly 0.999. However, the minimum value remains close to that observed at the smallest resolution. At the highest tested resolution, 128×128 , significantly higher AUC scores are achieved compared to the lower resolutions. Despite this improvement, there is an outlier among the five folds, which impacts the Glioma class the most and the Pituitary tumor class the least. Table 4.12 shows the overall confusion matrices for the baseline VGG model using its four best-performing configurations, as well as those for the L-net architecture at image sizes ranging from 16×16 to 128×128 . It is clear that the L-net model outperforms the baseline, with 40, 32, 20, and 10 misclassifications across the four image sizes, compared to 53 mistakes made by the best-performing baseline model on the 3064 tumor images. The number of correctly identified Meningioma cases by the baseline model matches the L-net’s performance at the 32×32 resolution, but at higher resolutions, L-net makes fewer mistakes. For the other two classes, L-net, at any tested resolution, consistently detects more tumors correctly than the baseline model in all configurations.

Table 4.12: Overall confusion matrices obtained by the best four VGG models [C7] (upper row), and the proposed L-net architecture at various image resolutions.

		Predicted			Predicted			Predicted			Predicted		
	Class	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary
		Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary
Actual	Meningioma	691	10	7	681	17	10	681	19	8	685	13	10
	Glioma	24	1399	3	25	1401	0	25	1401	0	29	1396	1
	Pituitary	5	4	921	3	2	925	3	2	925	3	3	924
		VGG first			VGG second			VGG third			VGG fourth		
		Predicted			Predicted			Predicted			Predicted		
	Class	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary
		Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary	Meningioma	Glioma	Pituitary
Actual	Meningioma	683	21	4	691	12	5	695	9	4	704	4	0
	Glioma	12	1414	0	11	1415	0	5	1421	0	4	1422	0
	Pituitary	1	2	927	3	1	926	1	1	928	1	1	928
		16×16			32×32			64×64			128×128		

Figure 4.27 presents examples of input images, one for each scenario from the overall confusion matrix where applicable, illustrating the classification results of the best-performing L-net model operating at a resolution of 128×128 .

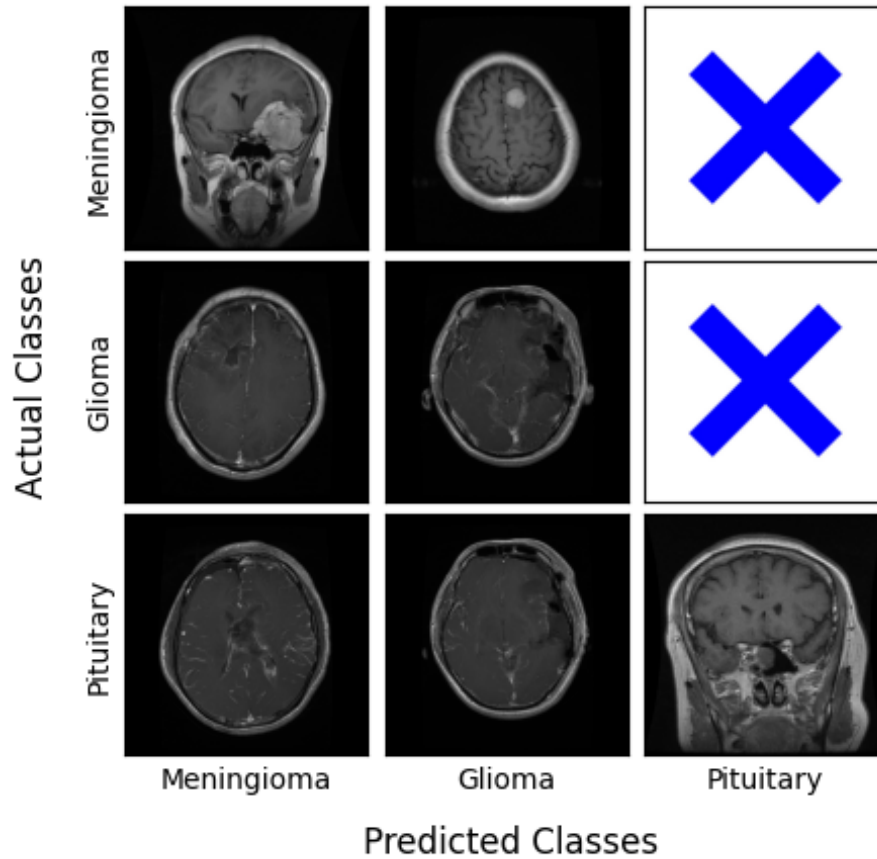


Figure 4.27: Some examples of correct and mistaken decisions made by the L-net architecture at the image size 128×128 , organized in the format of a confusion matrix. The Pituitary class has no false positives.

Overall, I have demonstrated that the proposed approach is effective. Using a U-net to preprocess or transform input images into intermediate data, and then feeding this data into a classifier, significantly improves decision accuracy. This is evidenced by the evaluation results presented earlier. Furthermore, I have shown that the L-net model can perform well even with smaller image sizes compared to CNN-based classification.

Table 4.10 presents the average results for different metrics and their standard deviations, indicating that even at a resolution of 64×64 , the average F1-score exceeds 0.99. Additionally, at 16×16 resolution, the results surpass those of my previous solution. Notably, the variance consistently decreases with the proposed architecture, as illustrated in Figure 4.25. Higher image resolutions result in lower benchmark variance, reflecting a more stable learning process.

Table 4.11 highlights the AUC metrics for each class, with particularly outstanding results for the Pituitary class, though its best performance is not achieved at the highest resolution. For the other classes, however, the 128×128 resolution yields the best results.

The confusion matrices, shown in Table 4.12, clearly illustrate the superior classification performance of the L-net model, with the total number of misclassifications reduced by up to 80% compared to previous solutions. In my best L-net model, the Pituitary tumor class has no false positives and only a few false negatives. While there are more misclassifications in the other two classes, the error rates are still significantly lower than earlier solutions.

These results represent a significant improvement, supporting the use of a U-net for feature extraction in a CNN-based classification model. Our method achieves high accuracy in tumor classification and also provides segmentation capabilities. This network structure is versatile and can be used for classifying and detecting various diseases, particularly in cases where multiple

output classes exist within a segmented region, such as Glioma, Meningioma, and Pituitary tumors. However, adapting the network for other datasets will require retraining and possibly adjusting the cost function.

4.4.9 Conclusion

In this research, I proposed a novel architecture for the classification of MRI tumor images, which simultaneously performs tumor segmentation. The network consists of two main components: first, a slightly modified U-net is used to segment the tumor from the input image, with the output of the U-net’s penultimate layer serving as an extracted feature. This feature is then passed to the second part of the architecture, a modified VGG network. Compared to previous methods, my approach improves classification accuracy from 98.3% to 99.6%, reducing misclassified cases by up to 80%.

In previous work, a traditional VGG model achieved the best classification accuracy at a 256×256 resolution. However, my model outperforms earlier models even with images as small as 16×16 . Nevertheless, using a 128×128 resolution is recommended for optimal classification performance. The L-net model exemplifies how training a neural network on two tasks simultaneously—classification and segmentation—can lead to better results than addressing them separately.

The results show a significantly lower error rate, as reflected in the confusion matrix. Additionally, the tables and boxplots demonstrate that the training process is highly stable at 128×128 resolution, with minimal fluctuations, though outliers are present in some metrics.

Future work will focus on validating the L-net architecture across various medical image datasets, a challenging task due to the limited availability of public datasets containing both annotated segmentation and classification labels.

4.5 Thesis Summary – MRI Research

This chapter presents my contributions to medical image processing using magnetic resonance imaging, covering two distinct research areas: infant brain tissue segmentation and brain tumor classification. Both directions involved the design and development of deep learning models with a focus on architectural innovation and performance optimization. The research resulted in several novel methodologies that significantly improved segmentation and classification accuracy, as validated through extensive experimentation.

Infant Brain Tissue Segmentation

The first part of this chapter focuses on the segmentation of infant brain tissues—namely cerebrospinal fluid (CSF), gray matter (GM), white matter (WM)—from MRI scans. In my first study [C8], I proposed a novel modification to the conventional U-net architecture: instead of increasing the number of filters as the network depth increased, I designed a version where the number of filters decreased. This approach led to a more compact and computationally efficient network without compromising performance.

In my second study [J5], I explored both a larger 2D U-net and a compact 3D U-net model. Although the 3D U-net used significantly fewer filters, it performed competitively with the 2D version, emphasizing the benefits of leveraging spatial context in volumetric data. I also introduced a novel composite loss function that combined Categorical Cross Entropy and Dice loss, each weighted equally (0.5). This hybrid cost function resulted in superior performance, with segmentation accuracy reaching an average Dice Similarity Coefficient (DSC) of 0.891 across tissue classes—outperforming baseline models that used only standard loss functions.

The third and most impactful innovation was the use of (2+1)D convolutions to model 3D MR volumes as sequences, drawing inspiration from video classification networks. This architecture

outperformed all previous models, providing the best segmentation results in terms of both Dice score and generalization. The novelty lies in applying spatiotemporal convolutional modeling to medical image segmentation, which is still an emerging approach in this field.

Brain Tumor Classification

The second research focus was brain tumor classification. Unlike traditional approaches that detect tumor presence, my work focused on determining the tumor type—specifically glioma, meningioma, or pituitary tumor—using T1-weighted MRI slices.

In my first study [C7], I developed a lightweight convolutional neural network and compared it against established state-of-the-art models (such as VGG variants). I tested multiple training strategies and found that selecting checkpoints based on maximum training accuracy led to more stable and better-performing models than monitoring minimum loss. Additionally, reducing the image resolution from 512×512 to 256×256 not only reduced computational cost but also improved classification performance. The network achieved an F1 score of 0.982, and the experiments revealed that dropout regularization was essential, while larger kernel sizes (e.g., 9×9) did not enhance performance. One key finding was that dense layer size had minimal influence on outcomes, which contrasts with common practices in CNN optimization.

The novelty of this study lies in showing that smaller, carefully optimized models can outperform deeper and more complex networks—an important consideration for real-time, resource-constrained environments.

In my final study [J6], I proposed a novel architecture by integrating the U-net segmentation network with the CNN-based classifier into a single multi-task learning framework. The final convolutional layer of the U-net was connected to the input of the CNN, allowing simultaneous segmentation and classification. This integration was made possible by the availability of tumor segmentation masks in the dataset. At an input resolution of 128×128 , the model achieved an F1 score exceeding 0.99—surpassing the best result (0.982) from earlier single-task models. This result validates the hypothesis that joint learning of spatial and semantic features can significantly improve performance. The proposed L-net architecture thus represents both a conceptual and practical innovation, capable of performing two diagnostic tasks within a single end-to-end trainable pipeline.

In summary, my MRI-focused research yielded novel and effective deep learning architectures for segmentation and classification. The achieved results—such as a Dice score of 0.891 for infant brain tissue segmentation and an F1 score over 0.99 for tumor classification—demonstrate state-of-the-art performance. The models are lightweight and suitable for real-world deployment on personal computers or cloud-based services, offering practical solutions to assist clinicians in accurate and efficient diagnosis.

5. Conclusion

This dissertation presented a comprehensive investigation into the application of machine learning methods to solve critical challenges in healthcare, with a particular emphasis on diabetes management and medical image analysis. The dual focus of the research—on physiological signal processing and image-based diagnostics—highlights the growing intersection between data-driven modeling and medical practice. Throughout this work, we demonstrated that machine learning is not only capable of modeling complex processes but can also be instrumental in automating decision-making systems that improve clinical outcomes.

In the domain of diabetes informatics, novel methodologies were introduced for detecting physical activity and food intake gestures using synthetic and real-world data from wearable sensors. These models, ranging from traditional classifiers to advanced recurrent neural networks, proved effective in identifying behavior patterns crucial for glucose regulation. Furthermore, we proposed and implemented a reinforcement learning-based framework for automated insulin control, showing that such systems can maintain blood glucose within a safe range, even in the face of variability in patient behavior and physiology. These contributions are not merely theoretical but lay the groundwork for next-generation, personalized artificial pancreas systems that can reduce the burden of disease management on patients.

The second major contribution of this thesis lies in the field of medical image processing. Through a series of carefully designed deep learning architectures, including 2D, 3D, and spatiotemporal U-Net variants, we achieved high-accuracy segmentation of infant brain tissues. Additionally, the introduction of the L-net—an integrated model combining segmentation and classification—enabled robust classification of glioma, meningioma, and pituitary tumors in MRI scans. The effectiveness of these methods was validated through extensive experimentation, demonstrating state-of-the-art performance and reinforcing the value of machine learning in enhancing radiological workflows and diagnostic precision.

Across all research areas, particular attention was paid to the validation of models using both synthetic and real clinical datasets. This pragmatic approach ensured the relevance and applicability of the proposed methods in real-world settings. The modular and reproducible nature of the solutions also ensures they can be extended or adapted to a wide range of medical applications in the future.

Ultimately, the work presented in this dissertation reflects a step forward in the development of intelligent systems for healthcare. By integrating domain knowledge with data-centric modeling, we have shown that it is possible to create robust, adaptive, and clinically meaningful solutions. The findings underscore the transformative potential of artificial intelligence in medicine and point toward a future where machine learning systems become integral components of both preventive and diagnostic care.

The interdisciplinary nature of the research—combining biomedical knowledge, sensor technologies, and machine learning—reflects the direction in which healthcare innovation is evolving. As data becomes increasingly central to clinical decision-making, the methodologies developed herein can play a key role in enabling more proactive, patient-specific, and efficient healthcare delivery systems.

This dissertation thus contributes to the broader vision of building AI-driven frameworks that do not simply automate processes, but enhance human understanding, improve outcomes, and ultimately empower both patients and healthcare professionals alike.

6. Future Research Directions

This chapter outlines future research directions based on the foundations laid by this dissertation. The proposed extensions aim to enhance the robustness, generalizability, and clinical relevance of the methods developed, while also opening new lines of inquiry. These suggestions are not presented as corrections to prior work, but rather as logical and valuable extensions informed by current trends in machine learning and medical informatics.

6.1 Detection of Physical Activity

The current models for physical activity detection successfully demonstrated the feasibility of integrating CGM and heart rate data using both classical and deep learning methods. In future work, extending this with additional sensor modalities (e.g., accelerometers or gyroscope data) could further improve recognition accuracy under varied physical conditions. Moreover, personalized modeling approaches could be explored more deeply to account for patient-specific physiological responses. While simple neural architectures were sufficient for this study, future work might incorporate attention-based temporal models or transformer-based networks to capture long-term dependencies more effectively.

6.2 Gesture Detection

The current system for gesture detection using inertial data from wearable sensors provides a promising foundation for recognizing carbohydrate intake events. Future research could involve expanding the gesture vocabulary, increasing subject diversity in training data, and refining real-time performance. Moreover, integrating additional contextual signals (such as meal timing or self-reports) may improve classification accuracy. Exploring alternative model architectures like temporal convolutional networks (TCNs) or lightweight transformer models may offer both efficiency and performance benefits. These directions aim to scale up the current system for deployment in real-world, patient-facing applications.

6.3 Reinforcement Learning-Based Insulin Regulation

The reinforcement learning framework proposed in this thesis demonstrated that data-driven policy and value networks can successfully regulate insulin dosing in simulated environments. To further this line of research, future work may focus on refining the virtual patient model with more physiological detail and inter-individual variability. Additionally, hybrid approaches combining reinforcement learning with control-theoretic models (e.g., MPC or PID tuning) may improve safety and convergence rates. While PPO was used as the learning algorithm in this study, future comparisons with other actor-critic methods such as A3C or SAC could help identify more efficient learning strategies.

6.4 Infant Brain Tissue Segmentation

The proposed U-Net variants provided robust segmentation of neonatal brain tissues. Future research could investigate the performance of alternative architectures such as TransU-net, which offer distinct advantages in terms of feature granularity and context-aware learning. These explorations would extend the current methodology, offering comparative insights and possibly hybridized architectures. Furthermore, incorporating spatial priors or anatomical atlases as regularization methods may improve segmentation quality, particularly in low-contrast regions. Additionally, longitudinal segmentation studies over time-series MRI scans could open the door to developmental trajectory analysis.

6.5 Brain Tumor Detection

The L-net framework presented in this dissertation successfully integrated segmentation and classification into a unified architecture. Future work could examine more granular tumor subtypes or expand to multi-sequence MRI data (e.g., FLAIR, T2) to capture broader diagnostic features. Additionally, exploring semi-supervised or self-supervised pretraining strategies could enable improved generalization, especially in low-data scenarios. Another promising direction involves explainability—using attention mechanisms or saliency maps to provide clinicians with interpretable outputs that support trust in AI-assisted diagnosis.

In all these areas, the continued integration of domain knowledge, patient-specific modeling, and advanced neural architectures will be key to translating research into clinically impactful tools.

7. Contribution in publications

- [C8] Conceptualization, Methodology, Writing, Visualization
- [J5] Conceptualization, Methodology, Writing, Visualization, Resource
- [C7] Methodology, Writing, Visualization
- [C5] Conceptualization, Methodology, Writing, Visualization
- [J1] Conceptualization, Methodology, Writing, Visualization, Software, Investigation, Data Curation
- [C4] Conceptualization, Methodology, Writing, Visualization, Validation
- [C3] Conceptualization, Methodology, Writing, Visualization, Investigation
- [C2] Conceptualization, Methodology, Writing, Visualization
- [C1] Conceptualization, Methodology, Writing, Visualization
- [J2] Conceptualization, Methodology, Writing, Visualization, Software, Investigation, Data Curation
- [C6] Conceptualization, Data Curation, Writing, Visualization
- [J3] Writing, Visualization, Validation, Resources, Methodology, Formal analysis, Data curation, Conceptualization.
- [J4] Conceptualization, Methodology, Validation, Writing, Visualization
- [J6] Methodology, Software, Validation, Writing, Visualization
- [J7] Conceptualization, Methodology, Validation, Resources, Data Curation, Writing, Visualization

8. Appendix

8.1 Physical activity supplementary

In addition to the F1-score discussed in the main text, an analysis was conducted to determine which parameter configurations are sufficient to achieve the desired performance. To accomplish this, various parameters are plotted as box plots alongside their values. The metrics used in this analysis include Accuracy, Precision, and Recall. These metrics were evaluated numerically for the top 30 models for both GRU and LSTM architectures and visualized using box plots. This section complements the F1 score results presented earlier in the Results section.

8.1.1 Precision

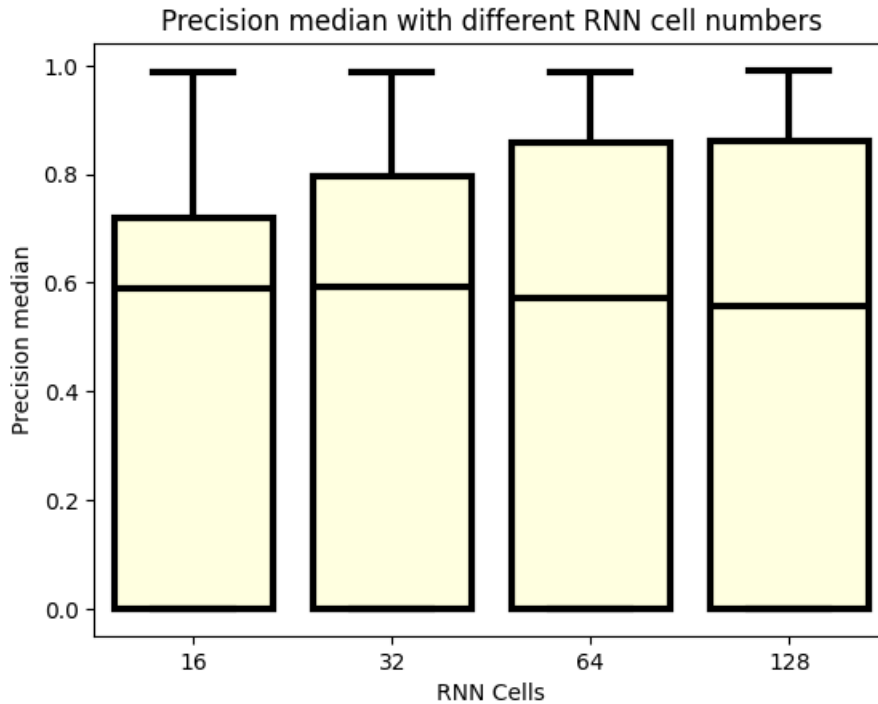


Figure 8.1: Precision value for different number off RNN cell

I start by examining the box plot results in Figure 8.1, which displays precision values for different RNN cell sizes (16, 32, 64, 128). The median precision values for each cell size are calculated and grouped according to other variables. Notably, all four configurations show instances of high performance based on precision metrics, with maximum values nearing one. However, it's important to note that the median precision value tends to decrease as the number of RNN cells increases. This can be explained by the gradient vanishing problem, where larger recurrent stacks lead to more significant information loss. Despite this, models with larger numbers of recurrent

cells generally perform better than those with only 16 cells. Moreover, the upper quartile values are higher for configurations with 32, 64, and 128 cells. However, performance starts to plateau for configurations with 64 and 128 cells. Thus, from a precision perspective, using 64 recurrent layers is sufficient for good performance. Additionally, using a higher number of cells results in longer learning times. Nonetheless, the number of RNN cells does not significantly impact the model's overall performance.

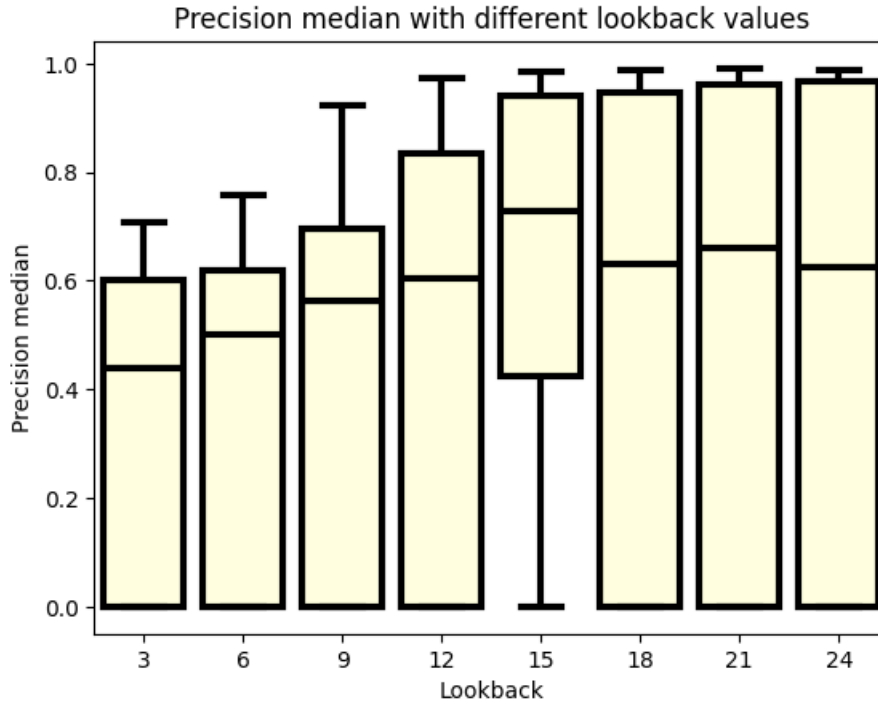


Figure 8.2: Precision value for different number off Look back

Let's examine how the look-back window affects the Precision metric, as shown in Figure 8.2. The median Precision value is calculated for each look-back window, ranging from 3 to 24. These values correspond to data observations taken at 5-minute intervals, resulting in a look-back window ranging from 15 minutes to two hours, with a 15-minute difference between each value.

The box plot reveals a noticeable staircase pattern, with the median values gradually increasing from a look-back value of 3 to 15. A similar trend is observed for the upper quartile and maximum values. However, beyond a look-back value of 15, no further improvements are noted. While the upper quartile values continue to rise, the median values begin to decline, and the maximum values level off. This indicates that a look-back value of 15 is sufficient for achieving good performance. Nonetheless, using the maximum look-back value of 24 might result in slightly better outcomes, though it would require longer learning times

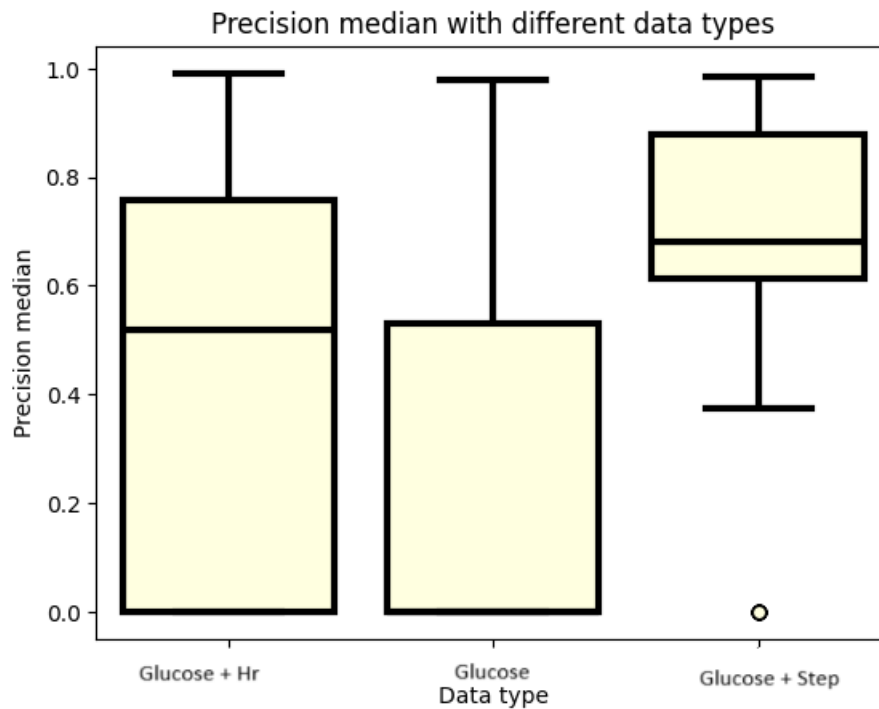


Figure 8.3: Precision value for different datatype

Figure 8.3 shows box plots comparing performance metrics based on the type of data used. Three scenarios are considered: using only blood glucose level data, using blood glucose level and heart rate data, and using blood glucose level and step count data. The analysis focuses on the median Precision values.

Initially, it is clear that relying solely on blood glucose level data is inadequate due to the model's delayed response to changes in blood glucose levels, as these changes are reflected with a lag. However, adding step count or heart rate data helps the model handle sudden changes until blood glucose level adjustments occur. Notably, using step count data results in better performance compared to heart rate data. Interestingly, the minimum Precision value is not zero when heart rate data is included. Overall, models perform best when incorporating step count data along with blood glucose level data.

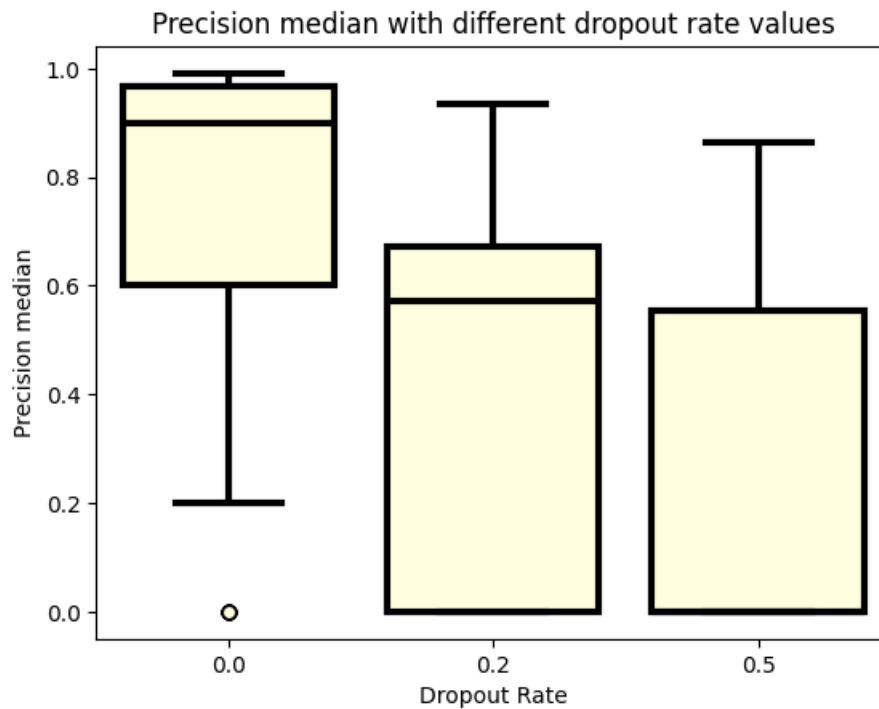


Figure 8.4: Precision value for different number off Drop out rate

Figure 8.4 illustrates the examination of different dropout rates, including 0.0, 0.2, and 0.5. Dropout rates are used to prevent overfitting, but excessively high dropout rates can hinder the model's ability to learn effectively from the dataset.

In this analysis, it is observed that using a dropout rate of 0.0, meaning no dropout, allows the model to achieve significantly high performance. In contrast, the introduction of dropout leads to a decline in model performance. Specifically, as dropout rates increase, the lower quartile values approach zero, indicating poorer performance. Moreover, for a dropout rate of 0.5, the median value is also near zero, further indicating reduced model effectiveness at higher dropout rates.

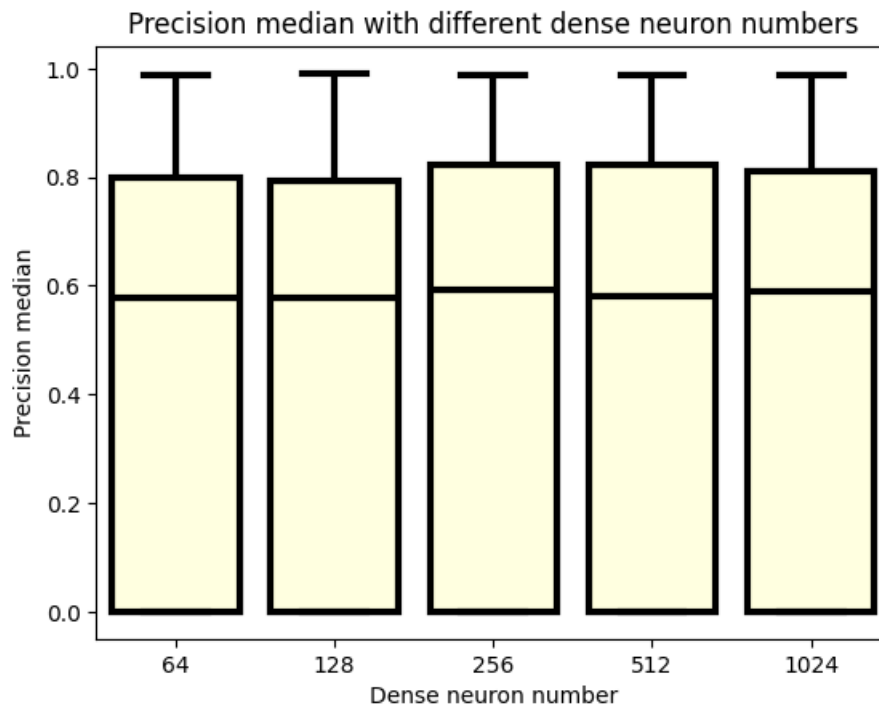


Figure 8.5: Precision value for different number off Dense neurons

Figure 8.5 shows the analysis of Precision values in relation to the number of neurons in the dense layer, ranging from 64 to 1024. The analysis of median Precision values reveals that this parameter has minimal or no noticeable effect on model performance. The median values show little variation across different neuron counts, with the exception of 256 neurons, which exhibits a relatively high value. Additionally, the upper quartile values are highest for the 256-neuron configuration, although the differences are not significantly greater than those in median values. Furthermore, the maximum Precision values remain consistent across all configurations, indicating that each neuron count has a configuration capable of achieving values close to 1. Thus, the number of neurons in the dense layer does not significantly impact the model's performance, as each configuration can achieve high Precision values.

8.1.2 Recall

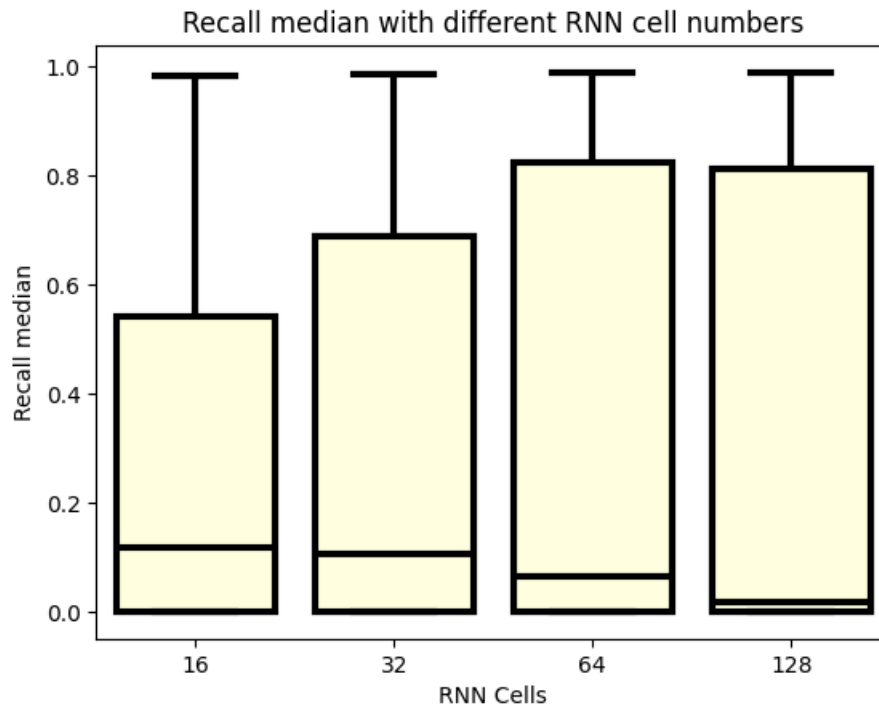


Figure 8.6: Recall value for different number off RNN cell

Figure 8.6 examines the Recall metric for different RNN cell sizes. While similar trends to those seen with Precision are observed, there are some notable differences. The median values in the box plots are generally lower for Recall than for Precision. A significant difference is also noted in the upper quartile, with the highest value for Recall occurring at 64 RNN cells. Despite these differences, it is clear that models with various numbers of RNN cells perform well, as each configuration can achieve Recall values close to 1. However, the results suggest that using more than 64 RNN cells does not significantly improve model performance.

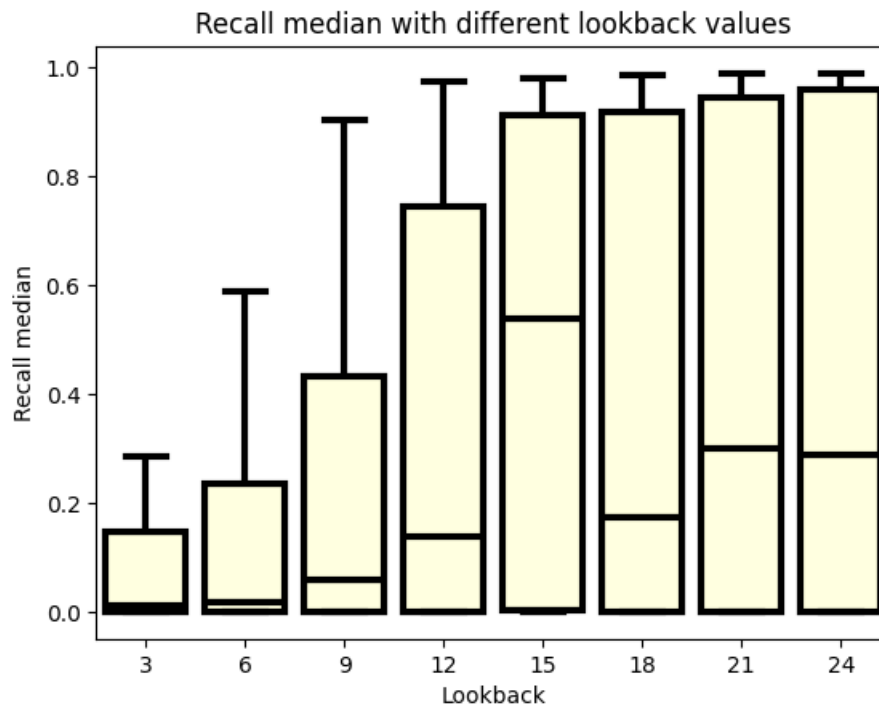


Figure 8.7: Recall value for different number off Look back

Figure 8.7 examines the Recall metric concerning the look-back time. There is noticeably greater variability compared to Precision, which is expected given that Recall metrics often have more false negatives than false positives, especially when there are fewer positive classes. Analysis of the graph reveals that smaller look-back values offer little benefit. Models begin to perform well with a look-back of 9, achieving Recall values above 0.8. However, most models struggle to reach 0.2, as indicated by median values below this threshold. The highest median value occurs at a look-back of 15, but there is a significant decrease in the median for larger look-back values compared to Precision. Nonetheless, using a look-back of 24 is more beneficial, as indicated by the highest upper quartile value and a maximum value approaching 1. Despite the longer learning time associated with higher look-back values, the performance gains outweigh the increased learning time.

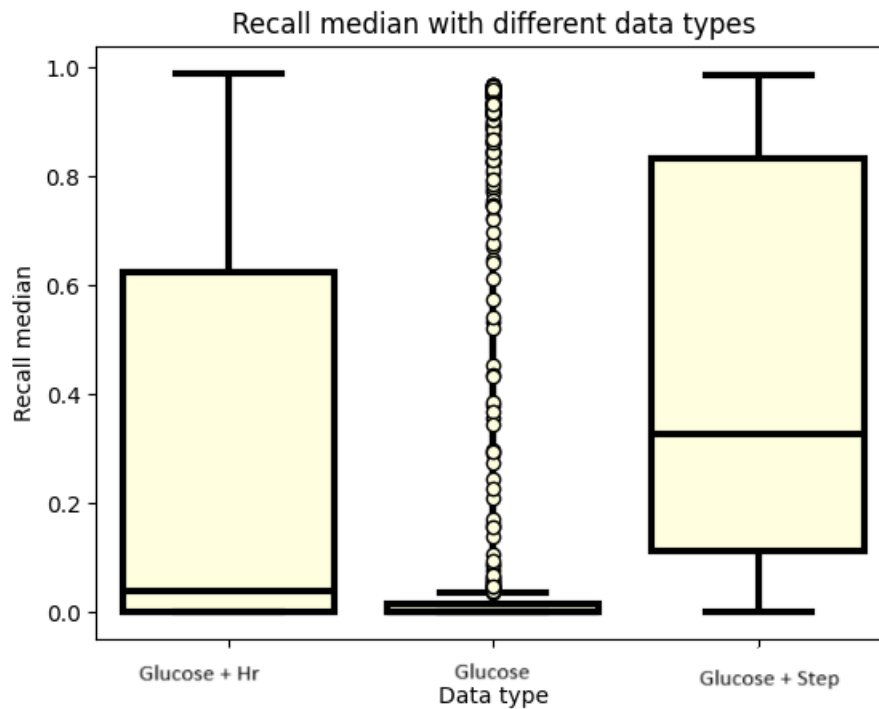


Figure 8.8: Recall value for different datatype

Figure 8.8 illustrates the different datasets and analyzes their respective Recall metric results. The datasets include blood glucose level alone, blood glucose level with heart rate, and blood glucose level with step count. The chart analysis reveals that relying solely on blood glucose data is generally suboptimal, as only a few outlier values approach 1 in the Recall metric. However, it is notable that some models achieve high performance using this data alone. When heart rate data is combined with blood glucose levels, the median Recall value remains below 0.2, indicating poor performance. Nevertheless, there are models that achieve maximum Recall values close to 1, although without outliers. The most effective dataset configuration is observed when step count data is included with blood glucose level data. This configuration has the highest median value among the three datasets, indicating superior performance. Additionally, the upper quartile value exceeds a Recall of 0.8, meaning that 25% of the models achieve a Recall greater than 0.8. However, this performance improvement is specific to this particular data configuration.

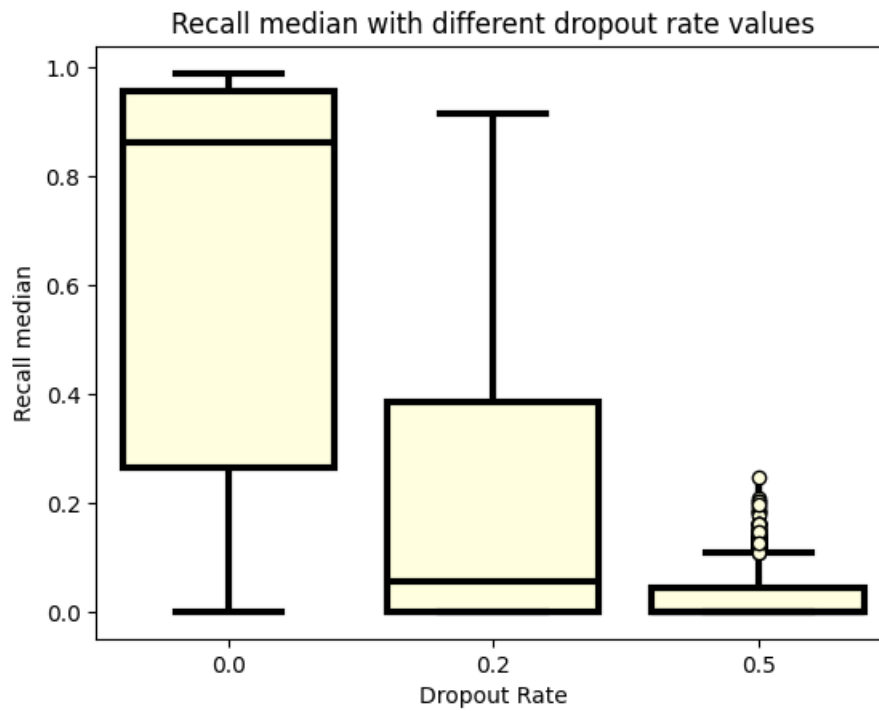


Figure 8.9: Recall value for different number off Drop out rate

Figure 8.9 examines the impact of different dropout rates on the Recall metric. A dropout rate of 0.5 shows the poorest performance, with no model configuration reaching a Recall value of 0.4. Similarly, a dropout rate of 0.2 produces suboptimal results, though slightly better than a 0.5 dropout rate. While the maximum Recall value reaches 0.9 with a 0.2 dropout rate, it is still inferior to the performance with a dropout rate of 0.0. It is clear that a dropout rate of 0.0 consistently delivers the best results, with a median Recall value exceeding 0.8. Additionally, both the upper quartile and maximum values can reach 1, indicating optimal model performance.

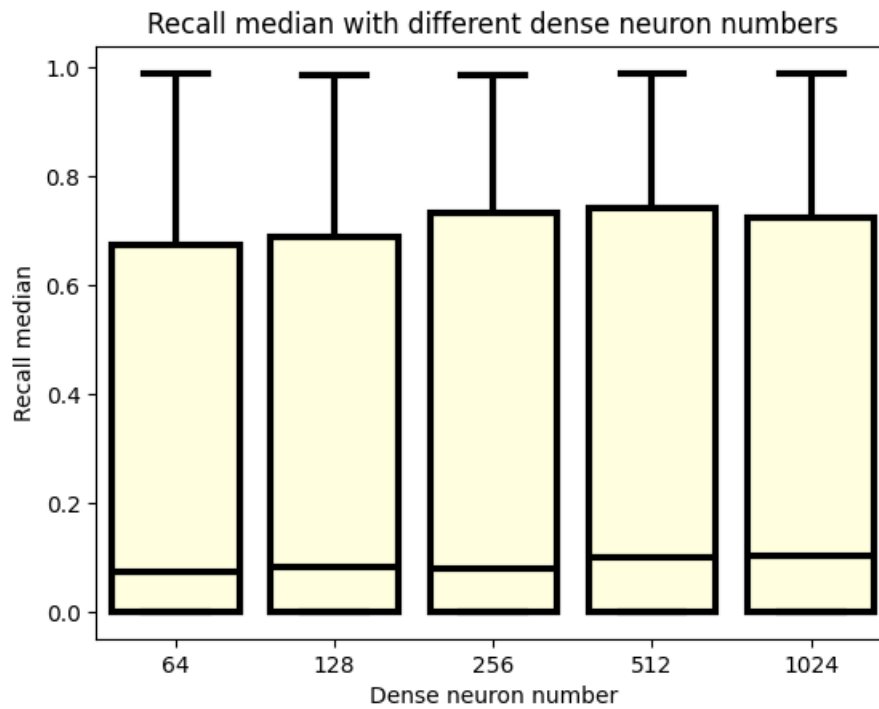


Figure 8.10: Recall value for different number off Dense neurons

Figure 8.10 plots the dense layer neuron counts based on the Recall metrics. Although there is slightly more variability compared to the Precision case, the differences are not statistically significant. Other parameters have a more substantial impact on the results. Using 256 or 512 neurons in the hidden layers may be slightly more beneficial, but these configurations do not significantly outperform others. In all cases, there is a configuration capable of achieving a maximum Recall value close to 1.

8.1.3 Accuracy

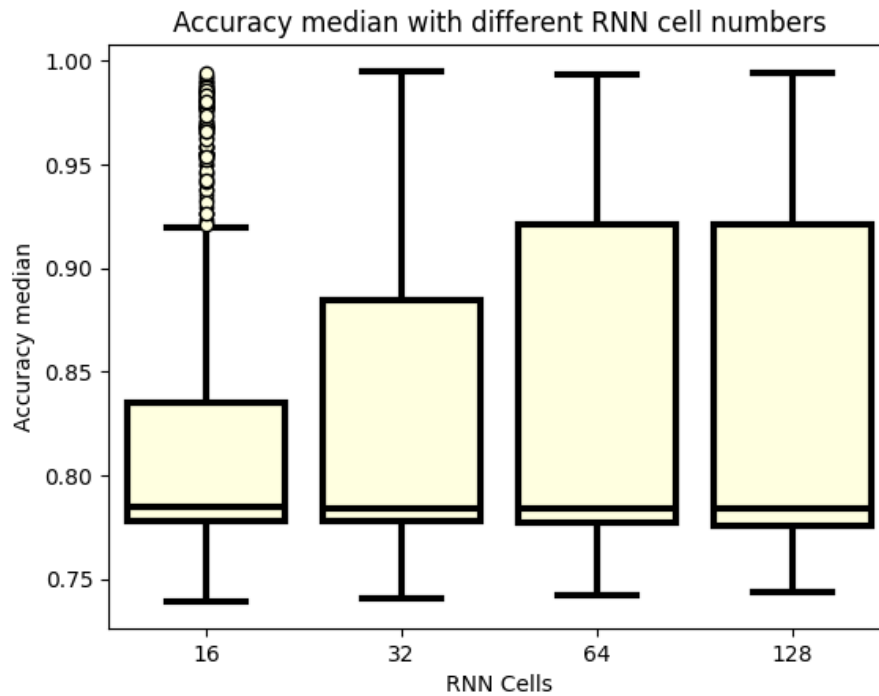


Figure 8.11: ACC value for different number off RNN cell

Figure 8.11 presents a plot of precision as a function of the number of RNN cells, along with their corresponding box plots. Unlike the consistent patterns observed in Precision, Recall, and F1 score values, the accuracy metric shows variations across different RNN cell counts. Models using 16 cells perform significantly worse than those with larger cell counts. In particular, models with 64 and 128 cells demonstrate superior performance, with upper quartile values exceeding 95%. Therefore, using these two cell numbers is recommended to achieve higher accuracy.

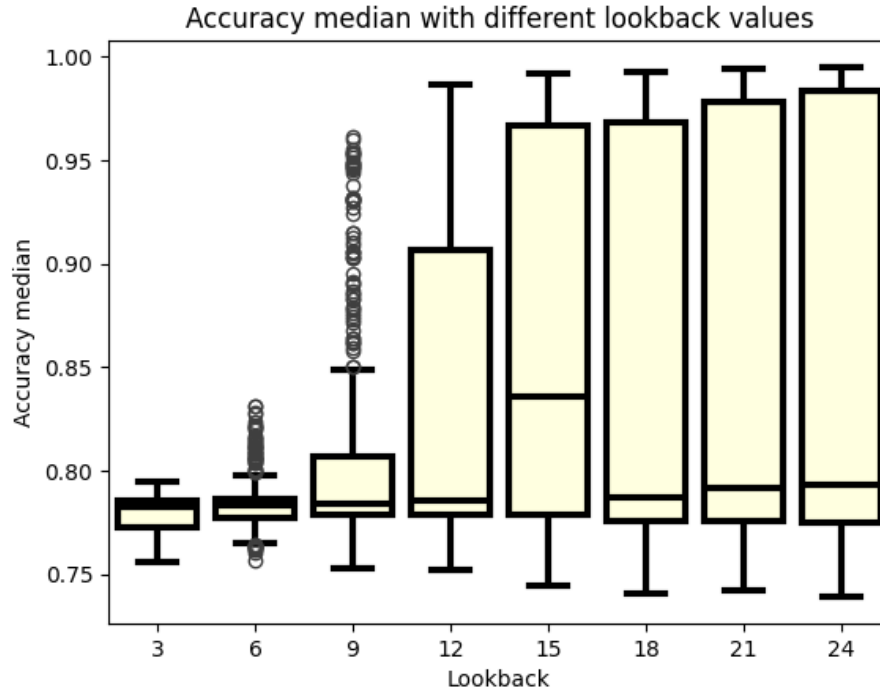


Figure 8.12: ACC value for different number off looc back

Figure 8.12 shows the different look-back windows ranging from 15 minutes to two hours. As the look-back window size increases, there is a noticeable improvement in performance. Outlier values exceed 80% for the 6th look-back, while for the 9th look-back, outlier values reach approximately 85% and can achieve results better than 95%. At the 12th look-back, no outliers are displayed, yet an outlier still reaches above 90%. A significant improvement is seen at the 15th look-back, where the upper quartile reaches 95%, and the best median value is achieved. As larger look-back windows are used, there are slight improvements in the upper quartile values. Interestingly, as performance improves, the minimum values decrease, indicating that while larger look-back windows yield better results, proper adjustment of other parameters is crucial. Based on accuracy, utilizing a 24-hour look-back window is advantageous.

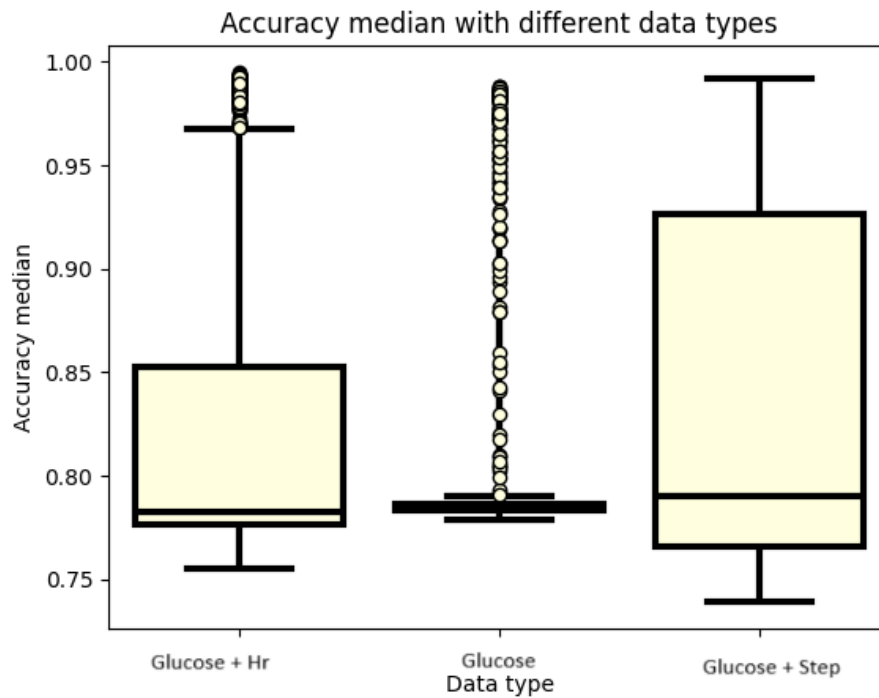


Figure 8.13: ACC value for different datatype

Figure 8.13 plots accuracy for different feature datasets. Using only blood glucose levels results in the poorest performance. However, similar to F1 score, Precision, and Recall, there are outlier configurations where good results can be achieved. The second-best feature set is when blood glucose levels and heart rate values are used. Unlike other metrics, outlier values are evident here. The best feature set is when both blood glucose levels and step counts are utilized. In this case, there are no outliers, and the median accuracy is the highest. Additionally, the upper quartile exceeds 90% only in this configuration.

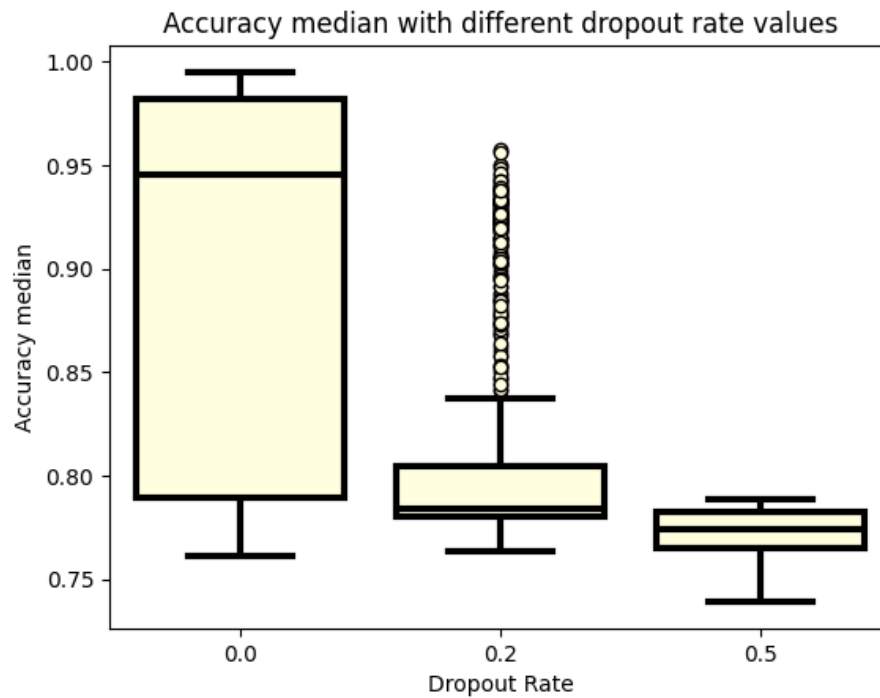


Figure 8.14: ACC value for different number off Drop out rate

Figure 8.14 examines the impact of dropout rates on accuracy. Similar to the cases of Precision, Recall, and F1 score, the worst performance is observed when dropout rates are applied. Conversely, the best performance is achieved without dropout rates. This is evident in the box plots, which show that models without dropout rates have superior accuracy. Although outliers with a dropout rate of 0.2 can achieve accuracy exceeding 95%, this is not true for a dropout rate of 0.5. Furthermore, when no dropout rate is used (dropout rate of 0.0), the median accuracy can reach nearly 95%.

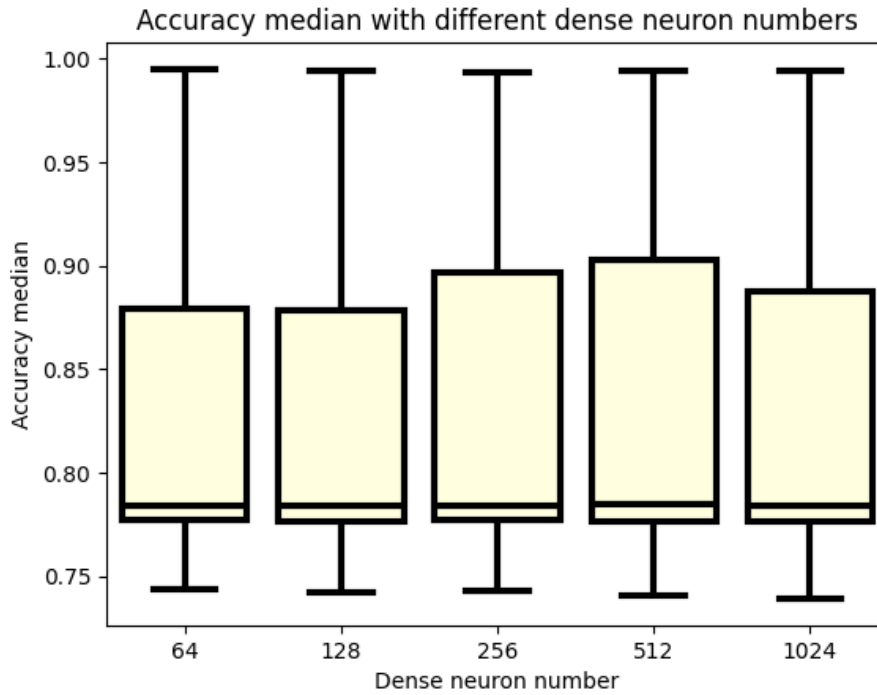


Figure 8.15: ACC value for different number off Dense neurons

Figure 8.15 examines the impact of different dense layer neuron counts on accuracy. Similar to the other metrics studied, there is no significant deviation in performance trends. For all neuron counts, there is a model capable of achieving an accuracy of 1. Notably, neuron counts of 256 and 512 appear to stand out, although overall, this parameter has the least impact on model performance.

8.1.4 Analyzation of the best 30 models

In this subsection, I ranked the top 30 models based on their F1 scores. The F1-score criterion provides a robust evaluation metric that balances precision and recall, ensuring that the selected models perform well across both aspects of classification accuracy. Due to the large size of the data, I divided the results into three tables. The top 30 models were ranked according to the median F1 score across five test cases. This ranking is presented in Table 8.1, which illustrates the accuracy of the AUC metric. Table 8.2 displays the Precision and Recall, while Table 3.8 shows the F1 score values.

Let's begin the analysis with Table 8.1, which presents models with different parameter settings. This table focuses on two metrics: AUC and precision. The models are listed in descending order based on their F1 scores. Most models use LSTM-based networks and incorporate both blood glucose and heart rate data. While the majority of models have a look-back window of 24, two models use a window length of 21. Interestingly, among the top thirty, there is a model with a look-back window of 18. As for the drop out rate, it is quite clear that all models in the best 30 used a dropout rate of zero. Regarding RNN cells, most models use either 128 or 64 cells, with some employing 32 cells, suggesting that this parameter may have a limited impact on model performance. The number of neurons in the dense layer appears to have the least influence, as the top thirty models utilize neuron counts of 64, 128, 256, 512, and 1024.

Examining the AUC metrics in the table reveals that all 30 models exhibit highly favorable results, with AUC values ranging from 0.998 to 0.999. The median value closely aligns with the mean, indicating minimal variance across the models, as supported by the standard deviation (STD) column, where the maximum standard deviation for the best model is 0.002.

Model	Data Type	Window Size	Dropout Rate	RNN Cells	Dense Neurons	Precision			Recall		
						Mean	Median	STD	Mean	Median	STD
LSTM	Glucose and HR	24	0	32	64	0.9981	0.9985	0.0020	0.9930	0.9949	0.0026
LSTM	Glucose and HR	21	0	128	128	0.9994	0.9995	0.0002	0.9944	0.9942	0.0011
LSTM	Glucose and HR	24	0	16	1024	0.9994	0.9996	0.0005	0.9927	0.9940	0.0031
LSTM	Glucose and HR	24	0	128	512	0.9994	0.9993	0.0004	0.9930	0.9940	0.0023
GRU	Glucose and HR	24	0	128	1024	0.9997	0.9997	0.0002	0.9942	0.9936	0.0025
LSTM	Glucose and HR	24	0	64	64	0.9993	0.9993	0.0005	0.9925	0.9936	0.0020
GRU	Glucose and HR	24	0	128	256	0.9992	0.9993	0.0001	0.9929	0.9932	0.0021
LSTM	Glucose and HR	21	0	128	64	0.9995	0.9996	0.0003	0.9933	0.9930	0.0005
LSTM	Glucose and HR	24	0	128	64	0.9996	0.9997	0.0004	0.9935	0.9932	0.0020
GRU	Glucose and HR	24	0	128	64	0.9996	0.9997	0.0003	0.9932	0.9928	0.0016
LSTM	Glucose and HR	21	0	128	1024	0.9994	0.9993	0.0003	0.9926	0.9930	0.0024
LSTM	Glucose and HR	24	0	64	128	0.9989	0.9986	0.0006	0.9920	0.9932	0.0022
LSTM	Glucose and Steps	24	0	128	128	0.9992	0.9991	0.0004	0.9919	0.9921	0.0020
LSTM	Glucose and HR	24	0	16	256	0.9990	0.9991	0.0007	0.9918	0.9928	0.0022
LSTM	Glucose and HR	18	0	128	256	0.9995	0.9996	0.0003	0.9929	0.9927	0.0007
LSTM	Glucose and HR	24	0	128	128	0.9997	0.9997	0.0001	0.9929	0.9932	0.0011
LSTM	Glucose and Steps	24	0	128	256	0.9992	0.9995	0.0007	0.9917	0.9921	0.0008
GRU	Glucose and HR	24	0	128	128	0.9992	0.9992	0.0004	0.9926	0.9928	0.0013
GRU	Glucose and Steps	24	0	128	128	0.9995	0.9994	0.0002	0.9918	0.9921	0.0027
LSTM	Glucose and HR	24	0	64	256	0.9991	0.9994	0.0007	0.9920	0.9923	0.0021
LSTM	Glucose and HR	24	0	128	256	0.9992	0.9993	0.0006	0.9927	0.9923	0.0013
LSTM	Glucose and HR	24	0	32	256	0.9989	0.9990	0.0007	0.9911	0.9928	0.0036
LSTM	Glucose and HR	24	0	32	512	0.9993	0.9991	0.0005	0.9933	0.9923	0.0020
LSTM	Glucose and Steps	24	0	128	64	0.9991	0.9992	0.0002	0.9915	0.9912	0.0014
GRU	Glucose and HR	24	0	64	256	0.9995	0.9996	0.0002	0.9918	0.9923	0.0012
LSTM	Glucose and HR	21	0	128	512	0.9988	0.9989	0.0008	0.9906	0.9921	0.0042
GRU	Glucose and Steps	24	0	128	64	0.9992	0.9993	0.0004	0.9912	0.9917	0.0020
GRU	Glucose and HR	24	0	64	512	0.9993	0.9992	0.0003	0.9917	0.9923	0.0018
GRU	Glucose and HR	21	0	64	1024	0.9988	0.9995	0.0015	0.9904	0.9921	0.0041
LSTM	Glucose and HR	24	0	32	128	0.9991	0.9992	0.0005	0.9923	0.9923	0.0015

Table 8.1: The 30 best model AUC and ACC scores

Model	Data Type	Window Size	Dropout Rate	RNN Cells	Dense Neurons	Precision			Recall		
						Mean	Median	STD	Mean	Median	STD
LSTM	Glucose and HR	24	0	32	64	0.9871	0.9873	0.0026	0.9816	0.9865	0.0109
LSTM	Glucose and HR	21	0	128	128	0.9913	0.9910	0.0050	0.9841	0.9853	0.0030
LSTM	Glucose and HR	24	0	16	1024	0.9821	0.9801	0.0082	0.9857	0.9814	0.0090
LSTM	Glucose and HR	24	0	128	512	0.9828	0.9868	0.0071	0.9865	0.9869	0.0058
GRU	Glucose and HR	24	0	128	1024	0.9869	0.9868	0.0068	0.9877	0.9888	0.0047
LSTM	Glucose and HR	24	0	64	64	0.9846	0.9833	0.0058	0.9825	0.9830	0.0055
GRU	Glucose and HR	24	0	128	256	0.9822	0.9797	0.0091	0.9858	0.9848	0.0042
LSTM	Glucose and HR	21	0	128	64	0.9850	0.9817	0.0073	0.9855	0.9871	0.0050
LSTM	Glucose and HR	24	0	128	64	0.9867	0.9865	0.0046	0.9844	0.9865	0.0064
GRU	Glucose and HR	24	0	128	64	0.9868	0.9878	0.0032	0.9839	0.9840	0.0043
LSTM	Glucose and HR	21	0	128	1024	0.9865	0.9892	0.0085	0.9817	0.9804	0.0048
LSTM	Glucose and HR	24	0	64	128	0.9809	0.9828	0.0077	0.9832	0.9845	0.0049
LSTM	Glucose and Steps	24	0	128	128	0.9865	0.9860	0.0041	0.9814	0.9815	0.0052
LSTM	Glucose and HR	24	0	16	256	0.9828	0.9818	0.0061	0.9813	0.9798	0.0076
LSTM	Glucose and HR	18	0	128	256	0.9867	0.9858	0.0039	0.9820	0.9806	0.0050
LSTM	Glucose and HR	24	0	128	128	0.9843	0.9850	0.0054	0.9837	0.9817	0.0057
LSTM	Glucose and Steps	24	0	128	256	0.9839	0.9833	0.0045	0.9831	0.9834	0.0053
GRU	Glucose and HR	24	0	128	128	0.9803	0.9813	0.0042	0.9873	0.9868	0.0032
GRU	Glucose and Steps	24	0	128	128	0.9848	0.9851	0.0034	0.9824	0.9814	0.0110
LSTM	Glucose and HR	24	0	64	256	0.9871	0.9868	0.0076	0.9784	0.9778	0.0032
LSTM	Glucose and HR	24	0	128	256	0.9847	0.9854	0.0062	0.9832	0.9849	0.0081
LSTM	Glucose and HR	24	0	32	256	0.9812	0.9807	0.0091	0.9788	0.9790	0.0092
LSTM	Glucose and HR	24	0	32	512	0.9872	0.9852	0.0043	0.9830	0.9816	0.0069
LSTM	Glucose and Steps	24	0	128	64	0.9852	0.9856	0.0076	0.9815	0.9806	0.0070
GRU	Glucose and HR	24	0	64	256	0.9860	0.9853	0.0031	0.9788	0.9761	0.0051
LSTM	Glucose and HR	21	0	128	512	0.9800	0.9866	0.0109	0.9773	0.9772	0.0100
GRU	Glucose and Steps	24	0	128	64	0.9843	0.9832	0.0034	0.9802	0.9832	0.0056
GRU	Glucose and HR	24	0	64	512	0.9770	0.9776	0.0058	0.9861	0.9887	0.0061
GRU	Glucose and HR	21	0	64	1024	0.9809	0.9823	0.0105	0.9767	0.9754	0.0109
LSTM	Glucose and HR	24	0	32	128	0.9874	0.9869	0.0015	0.9779	0.9793	0.0087

Table 8.2: The 30 best model Precision and Recall scores

In the accuracy column, the mean, median, and standard deviation are plotted similarly. Interestingly, although the second model has a higher average accuracy than the first, the median accuracy of the first model surpasses that of the second. Notably, the first model, considered the best, shows a much higher standard deviation, as indicated by the STD column, where the second model has half the value of the first. Despite this variance, minimal differences are observed between the top models. It is possible that the first model's superiority stems from its ability to learn the data better in a two-training case due to its smaller size, though this observation may not be generalizable.

8.2 Reinforcement learning based insulin control supplementary

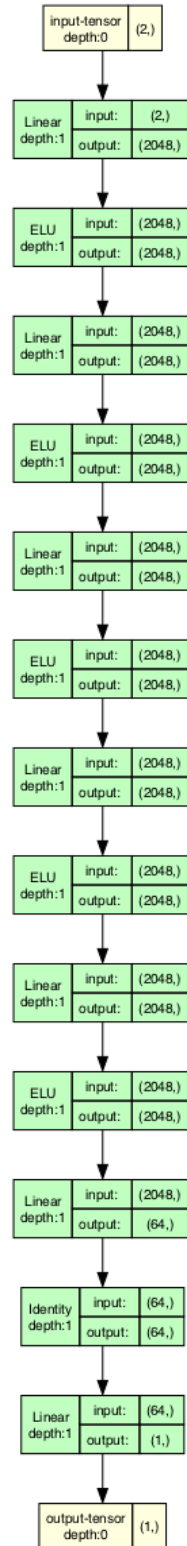


Figure 8.16: Value network

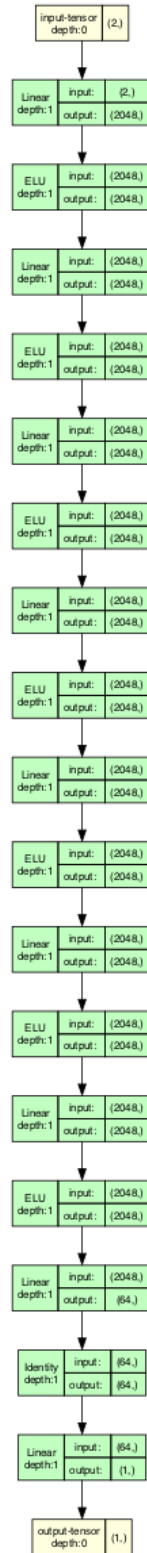


Figure 8.17: Policy network

Bibliography

- [R1] American Diabetes Association. “Diagnosis and classification of diabetes mellitus”. en. In: *Diabetes Care* 33 Suppl 1.Suppl 1 (Jan. 2010), S62–9.
- [R2] Dr. Pánczél Pál. “A cukorbetegség etiológiája és patogenezeise”. In: (2017). URL: <https://semmelweis.hu/belgyogyaszat3/files/2017/05/A-diabetes-mellitus-etiol%5C%c3%5C%b3gi%5C%c3%5C%a1ja-%5C%c3%5C%a9s-patogenezeise.pdf>.
- [R3] Anjali D Deshpande, Marcie Harris-Hayes, and Mario Schootman. “Epidemiology of diabetes and diabetes-related complications”. en. In: *Phys Ther* 88.11 (Sept. 2008), pp. 1254–1264.
- [R4] WHO. “Diabetes”. In: (2021). URL: <https://www.who.int/news-room/fact-sheets/detail/diabetes>.
- [R5] Xiling Lin, Yufeng Xu, Xiaowen Pan, Jingya Xu, Yue Ding, Xue Sun, et al. “Global, regional, and national burden and trend of diabetes in 195 countries and territories: an analysis from 1990 to 2025”. en. In: *Sci Rep* 10.1 (Sept. 2020), p. 14790.
- [R6] Medical Online. “A cukorbetegség nem csak az idősek betegsége”. In: (2019). URL: http://medicalonline.hu/gyogyitas/cikk/a_cukorbetegseg_nem_csak_az_idosek_betegsege.
- [R7] Eszter Vamos, Maria Kopp, András Keszei, Marta Novak, and Istvan Mucsi. “Prevalence of diabetes in a large, nationally representative population sample in Hungary”. In: *Diabetes research and clinical practice* 81 (July 2008), e5–8. DOI: 10.1016/j.diabres.2008.04.022.
- [R8] Prof. Dr. Wittmann István. “Diabetológia jegyzet orvostanhallgatók számára”. In: (2014). URL: <https://kk.pte.hu/hu/download/index/18362>.
- [R9] Dr. Gerő László. “A pancreatogen diabetes okai, tünettana és kezelése”. In: (). URL: <http://www.diabet.hu/folyoiratcikk.aspx?articleid=2505>.
- [R10] Ilka Gdanielcz. “Postprandial Blood Glucose - Why it’s important after eating”. In: (2018). URL: <https://www.mysugr.com/en/blog/postprandial-glucose-why-your-blood-sugar-is-so-important-after-eating/>.
- [R11] Researchfeatures. *Diabetes revolution: The new ‘artificial pancreas.’* Research Features. <https://researchfeatures.com/diabetes-revolution-new-artificial-pancreas/> (accessed on 31 Jan 2024). Jan. 2024.
- [R12] National High Magnetic Field Laboratory. *MRI: A Guided Tour - Magnet Academy*. <https://nationalmaglab.org/magnet-academy/read-science-stories/science-simplified/mri-a-guided-tour/>. n.d.
- [R13] P. C. LAUTERBUR. “Image Formation by Induced Local Interactions: Examples Employing Nuclear Magnetic Resonance”. In: *Nature* 242.5394 (Mar. 1973), pp. 190–191. ISSN: 1476-4687. DOI: 10.1038/242190a0. URL: <https://doi.org/10.1038/242190a0>.
- [R14] B. Kastler, Z. Patay, and K. Karlinger. *MRI orvosoknak*. Centauro, 1993. ISBN: 9788885980143. URL: <https://books.google.hu/books?id=eDenPQAACAAJ>.

- [R15] Aren Shah and Shima Aran. “A Review of Magnetic Resonance (MR) Safety: The Essentials to Patient Safety”. en. In: *Cureus* 15.10 (Oct. 2023), e47345.
- [R16] Kristin K Wenger, Kristina M Visscher, Francis M Miezin, Steven E Petersen, and Bradley L Schlaggar. “Comparison of sustained and transient activity in children and adults using a mixed blocked/event-related fMRI design”. en. In: *Neuroimage* 22.2 (June 2004), pp. 975–985.
- [R17] Wikipedia contributors. *Magnetic Resonance Imaging*. https://en.wikipedia.org/wiki/Magnetic_resonance_imaging. Last updated March 4, 2025. Accessed March 24, 2025. Mar. 2025.
- [R18] J. Assheuer and M. Sager. *MRI and CT Atlas of the Dog*. Blackwell science. Blackwell Science, 1997. ISBN: 9783894121648. URL: <https://books.google.hu/books?id=Zh1uQgAACAAJ>.
- [R19] Oona Rainio, Jarmo Teuho, and Riku Klén. “Evaluation metrics and statistical tests for machine learning”. In: *Scientific Reports* 14.1 (Mar. 2024), p. 6086.
- [R20] Charles X. Ling, Jin Huang, and Harry Zhang. “AUC: a statistically consistent and more discriminating measure than accuracy”. In: *Proceedings of the 18th International Joint Conference on Artificial Intelligence*. IJCAI’03. Acapulco, Mexico: Morgan Kaufmann Publishers Inc., 2003, pp. 519–524.
- [R21] Karimollah Hajian-Tilaki. “Receiver Operating Characteristic (ROC) Curve Analysis for Medical Diagnostic Test Evaluation”. en. In: *Caspian J Intern Med* 4.2 (2013), pp. 627–635.
- [R22] Oxana Trifonova, Petr Lokhov, and A.I. Archakov. “Metabolic profiling of human blood”. In: *Biomeditsinskaya Khimiya* 60 (May 2014), pp. 281–294. DOI: 10.18097/pbmc20146003281.
- [R23] Kai Ming Ting. “Confusion Matrix”. In: *Encyclopedia of Machine Learning*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2010, pp. 209–209. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_157. URL: https://doi.org/10.1007/978-0-387-30164-8_157.
- [R24] *How to interpret a confusion matrix for a machine learning model — evidentlyai.com*. <https://www.evidentlyai.com/classification-metrics/confusion-matrix>. [Accessed 27-03-2025].
- [R25] Boris Iglewicz. “Summarizing Data with Boxplots”. In: *International Encyclopedia of Statistical Science*. Ed. by Miodrag Lovric. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 1572–1575. ISBN: 978-3-642-04898-2. DOI: 10.1007/978-3-642-04898-2_582. URL: https://doi.org/10.1007/978-3-642-04898-2_582.
- [R26] Admin. *Box Plot (Definition, Parts, Distribution, Applications & Examples)*. BYJUS, <https://byjus.com/maths/box-plot/>. Mar. 2021.
- [R27] Lalo Magni, Davide M Raimondo, Chiara Dalla Man, Marc Breton, Stephen Patek, Giuseppe De Nicolao, et al. “Evaluating the efficacy of closed-loop glucose regulation via control-variability grid analysis”. en. In: *J Diabetes Sci Technol* 2.4 (July 2008), pp. 630–635.
- [R28] Virginia Bellido, Pedro José Pinés-Corrales, Rocío Villar-Taibo, and Francisco Javier Ampudia-Blasco. “Time-in-range for monitoring glucose control: Is it time for a change?” In: *Diabetes Research and Clinical Practice* 177 (2021), p. 108917. ISSN: 0168-8227. DOI: <https://doi.org/10.1016/j.diabres.2021.108917>. URL: <https://www.sciencedirect.com/science/article/pii/S0168822721002771>.
- [R29] Libby Evans. *Diabetes and Time in Range (TIR) — diabetesqualified.com.au*. <https://www.diabetesqualified.com.au/time-in-range/>. [Accessed 27-03-2025].

- [R30] Navid Resalat, Joseph El Youssef, Nichole Tyler, Jessica Castle, and Peter G Jacobs. “A statistical virtual patient population for the glucoregulatory system in type 1 diabetes with integrated exercise model”. en. In: *PLoS One* 14.7 (July 2019), e0217301.
- [R31] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. *Harnessing Deep Neural Networks with Logic Rules*. 2020. arXiv: 1603.06318 [cs.LG].
- [R32] Richard IG Holt, Clive Cockram, Allan Flyvbjerg, and Barry J Goldstein. *Textbook of diabetes*. Chichester, UK: John Wiley & Sons, 2017.
- [R33] S R Bird and J A Hawley. “Update on the effects of physical activity on insulin sensitivity in humans”. In: *BMJ Open Sport Exerc Med* 2 (2017), e000143.
- [R34] Pamela Martyn-Nemeth, Laurie Quinn, Sue Penckofer, Chang Park, Vanessa Hofer, and Larisa Burke. “Fear of hypoglycemia: Influence on glycemic variability and self-management behavior in young adults with type 1 diabetes”. In: *Journal of Diabetes and its Complications* 31 (2017), pp. 735–741.
- [R35] N Jeandidier, J P Riveline, N Tubiana-Rufi, A Vambergue, B Catargi, V Melki, et al. “Treatment of diabetes mellitus using an external insulin pump in clinical practice”. In: *Diabetes & Metabolism* 34 (2008), pp. 425–438.
- [R36] P O Adams. “The impact of brief high-intensity exercise on blood glucose levels”. In: *Diabetes Metab. Syndr. Obes.* 6 (2013), pp. 113–122.
- [R37] Lykke Sylow, Maximilian Kleinert, Erik A Richter, and Thomas E Jensen. “Exercise-stimulated glucose uptake—regulation and implications for glycaemic control”. In: *Nature Reviews Endocrinology* 13 (2017), pp. 133–148.
- [R38] Øyvind Stavadahl, Anders L Fougner, Konstanze Kölle, Sverre Chr Christiansen, Reinold Ellingsen, and Sven M Carlsen. “The artificial pancreas: A dynamic challenge”. In: *IFAC-PapersOnLine* 49 (2016), pp. 765–772.
- [R39] Sémah Tagougui, Nadine Taleb, Joséphine Molvau, Élisabeth Nguyen, Marie Raffray, and Rémi Rabasa-Lhoret. “Artificial pancreas systems and physical activity in patients with type 1 diabetes: challenges, adopted approaches, and future perspectives”. In: *Journal of diabetes science and technology* 13 (2019), pp. 1077–1090.
- [R40] Ulf Ekelund, Eric Poortvliet, Agneta Yngve, Anita Hurtig-Wennlöv, Andreas Nilsson, and Michael Sjöström. “Heart rate as an indicator of the intensity of physical activity in human adolescents”. In: *European journal of applied physiology* 85 (2001), pp. 244–249.
- [R41] C Crema, A Depari, A Flammini, Emiliano Sisinni, T Haslwanter, and S Salzmann. “IMU-based solution for automatic detection and classification of exercises in the fitness scenario”. In: *2017 IEEE Sensors Applications Symposium (SAS)*. 2017, pp. 1–6.
- [R42] Hoda Allahbakhshi, Timo Hinrichs, Haosheng Huang, and Robert Weibel. “The key factors in physical activity type detection using real-life data: a systematic review”. In: *Frontiers in physiology* 10 (2019), p. 75.
- [R43] Marzia Cescon, Divya Choudhary, Jordan E Pinsker, Vikash Dadlani, Mei Mei Church, Yogish C Kudva, et al. “Activity detection and classification from wristband accelerometer data collected on people with type 1 diabetes in free-living conditions”. In: *Computers in Biology and Medicine* 135 (2021), p. 104633.
- [R44] Alberto Verrotti, Giovanni Prezioso, Raffaella Scattoni, and Francesco Chiarelli. “Autonomic neuropathy in diabetes mellitus”. In: *Frontiers in endocrinology* (2014), p. 205.
- [R45] Shruti Agashe and Steven Petak. “Cardiac autonomic neuropathy in diabetes mellitus”. In: *Methodist DeBakey cardiovascular journal* 14 (2018), p. 251.

- [R46] Simon Helleputte, Tine De Backer, Bruno Lapauw, Samyah Shadid, Bert Celie, Birgit Van Eetvelde, et al. “The relationship between glycaemic variability and cardiovascular autonomic dysfunction in patients with type 1 diabetes: a systematic review”. In: *Diabetes/metabolism research and reviews* 36 (2020), e3301.
- [R47] Vini Vijayan, James P Connolly, Joan Condell, Nigel McKelvey, and Philip Gardiner. “Review of Wearable Devices and Data Collection Considerations for Connected Health”. In: *Sensors* 21 (2021), p. 5589.
- [R48] Liang Liang, Fan Wei Kong, Caitlin Martin, Thuy Pham, Qian Wang, James Duncan, et al. “Machine learning–based 3-D geometry reconstruction and modeling of aortic valve deformation using 3-D computed tomography images”. In: *Int J Numer Meth Biomed Engng* 33 (2017), e2827.
- [R49] Scott M Lundberg, Bala Nair, Monica S Vavilala, Mayumi Horibe, Michael J Eisses, Trevor Adams, et al. “Explainable machine-learning predictions for the prevention of hypoxaemia during surgery”. In: *Int J Numer Meth Biomed Engng* 2 (2018), pp. 749–760.
- [R50] S Ostrogonac, Edvin Pakoci, Milan Sečujski, and Dragiša Mišković. “Morphology-based vs unsupervised word clustering for training language models for Serbian”. In: *Acta Polytechnica Hungarica: Journal of Applied Sciences* (2019).
- [R51] Kristína Machová, Marian Mach, and Miroslava Hrešková. “Classification of Special Web Reviewers Based on Various Regression Methods”. In: *Acta Polytechnica Hungarica* 17 (2020).
- [R52] Peter Bednár, Juliana Ivančáková, and Martin Sarnovský. “Semantic Composition of Data Analytical Processes”. In: *Acta Polytechnica Hungarica* 21.2 (2024).
- [R53] Amir Hayeri. “Predicting Future Glucose Fluctuations Using Machine Learning and Wearable Sensor Data”. In: *Diabetes* 67 (2018), A193.
- [R54] Elena Daskalaki, Peter Diem, and Stavroula G Mougiakakou. “Model-free machine learning in biomedicine: Feasibility study in type 1 diabetes”. In: *PloS one* 11 (2016), e0158722.
- [R55] Ashenafi Zebene Woldaregay, Eirik Årsand, Taxiarchis Botsis, David Albers, Lena Mamykina, and Gunnar Hartvigsen. “Data-Driven Blood Glucose Pattern Classification and Anomalies Detection: Machine-Learning Applications in Type 1 Diabetes”. In: *Journal of medical Internet research* 21 (2019), e11030.
- [R56] Ivan Contreras and Josep Vehi. “Artificial intelligence for diabetes management and decision support: literature review”. In: *Journal of Medical Internet Research* 20 (2018), e10775.
- [R57] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors”. In: *Cognitive modeling* 5 (1988), p. 1.
- [R58] Mohammad Reza Askari, Mudassir Rashid, Xiaoyu Sun, Mert Sevil, Andrew Shahidehpour, Keigo Kawaji, et al. “Detection of Meals and Physical Activity Events From Free-Living Data of People With Diabetes”. In: *J Diabetes Sci Technol* 17.6 (June 2022), pp. 1482–1492.
- [R59] Yukun Zeng, Jianlei Zhang, and Binil Starly. “Recurrent neural networks with long term temporal dependencies in machine tool wear diagnosis and prognosis”. In: *SN Applied Sciences* 3 (Apr. 2021). DOI: 10.1007/s42452-021-04427-5.
- [R60] Navid Resalat, Joseph El Youssef, Nichole Tyler, Jessica Castle, and Peter G Jacobs. “A statistical virtual patient population for the glucoregulatory system in type 1 diabetes with integrated exercise model”. In: *PloS One* 14.7 (2019).

- [R61] Miriam M N Nærum. *Model Predictive Control for Insulin Administration in People with Type 1 Diabetes*. BSc Thesis, Kongens Lyngby. 2010.
- [R62] D Boiroux and J B Jørgensen. “A Nonlinear Model Predictive Control Strategy for Glucose Control in People with Type 1 Diabetes”. In: *IFAC-PapersOnline* 51-27 (2018), pp. 192–197.
- [R63] David R Cox. “The regression analysis of binary sequences (with discussion)”. In: *J R Stat Soc B* 20.2 (1958), pp. 215–242.
- [R64] Y Freund and R E Schapire. “Experiments with a new boosting algorithm”. In: *13th International Conference on Machine Learning*. 1996, pp. 148–157.
- [R65] S B Akers. “Binary decision diagrams”. In: *IEEE Transactions on Computers* C-27.6 (1978), pp. 509–516.
- [R66] L Szilágyi, D Iclănzan, Z Kapás, Zs Szabó, Á Györfi, and L Lefkovits. “Low and high grade glioma segmentation in multispectral brain MRI data”. In: *Acta Universitatis Sapientiae – Informatica* 10.1 (2018), pp. 110–132.
- [R67] C K Chow. “An optimum character recognition system using decision functions”. In: *IRE Transactions on Computers* EC-6 (1957), pp. 247–254.
- [R68] T M Cover and P E Hart. “Nearest neighbor pattern classification”. In: *IEEE Transactions on Information Theory* IT-13 (1967), pp. 21–27.
- [R69] Jon Louis Bentley. “Multidimensional binary search trees used for associative searching”. In: *Communications of the ACM* 18.9 (1975), pp. 509–517.
- [R70] Corinna Cortes and Vladimir Vapnik. “Support-vector networks”. In: *Machine Learning* 20 (1995), pp. 273–297.
- [R71] L Breiman. “Random forests”. In: *Machine Learning* 45.1 (2001), pp. 5–32.
- [R72] Esteban Alfaro-Cortés, José-Luis Alfaro-Navarro, Matías Gámez, and Noelia García. “Using random forest to interpret out-of-control signals”. In: *Acta Polytech. Hung.* 17.6 (2020), pp. 115–130.
- [R73] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors”. In: *Nature* 323.6088 (1986), pp. 533–536.
- [R74] O P Trifonova, P G Lokhov, and A I Archakov. “Metabolic profiling of human blood”. In: *Biochem. Moscow Suppl. Ser. B* 7 (2013), pp. 179–186.
- [R75] Cindy Marling and Razvan C Bunescu. “The OhioT1DM dataset for blood glucose level prediction”. In: *KHD@ IJCAI*. 2018.
- [R76] Oladimeji Farri, Aili Guo, Sadid Hasan, Zina Ibrahim, Cindy Marling, Jesse Raffa, et al. “Blood Glucose Prediction Challenge”. In: *The 3rd International Workshop on Knowledge Discovery in Healthcare Data, CEUR Workshop Proceedings (CEUR-WS.org)*. 2018.
- [R77] Kerstin Bach, Razvan C Bunescu, Cindy Marling, and Nirmalie Wiratunga. “Blood Glucose Prediction Challenge”. In: *The 5th International Workshop on Knowledge Discovery in Healthcare Data, 24th European Conference on Artificial Intelligence (ECAI 2020)*. 2020.
- [R78] Cindy Marling and Razvan Bunescu. “The ohiot1dm dataset for blood glucose level prediction: Update 2020”. In: *KHD@ IJCAI* (2020).
- [R79] Fabien Dubosson, Jean-Eudes Ranvier, Stefano Bromuri, Jean-Paul Calbimonte, Juan Ruiz, and Michael Schumacher. “The open D1NAMO dataset: A multi-modal dataset for research on non-invasive type 1 diabetes management”. In: *Informatics in Medicine Unlocked* 13 (2018), pp. 92–100.

- [R80] *Zephyr Bioharness 3.0 User Manual*. Available at <https://www.zephyranywhere.com/media/download/user-manual.pdf>. Zephyr Technology Inc. Version 9700.0079, 2012.
- [R81] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.
- [R82] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press, 2016.
- [R83] Kevin Alvarez, Julio Urenda, Orsoly Csiszár, Gábor Csiszár, József Dombi, György Eigner, et al. “Towards Fast and Understandable Computations: Which” And”-and” Or”-Operations Can Be Represented by the Fastest (ie, 1-Layer) Neural Networks? Which Activations Functions Allow Such Representations?” In: (2020).
- [R84] Cindy Marling Razvan Bunescu and Jay Shubbrook. “Blood Glucose Prediction Using Physiological Models and Support Vector Regression”. In: *Machine Learning for Health* (2012).
- [R85] Mike Schuster and Kuldip K Paliwal. “Bidirectional recurrent neural networks”. In: *IEEE Transactions on Signal Processing* 45 (1997), pp. 2673–2681.
- [R86] Alex Graves and Jürgen Schmidhuber. “Framewise phoneme classification with bidirectional LSTM and other neural network architectures”. In: *Neural Networks* 5-6 (2005), pp. 602–610.
- [R87] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: 1502.03167 [cs.LG].
- [R88] Gousia Habib and Shaima Qureshi. “GAPCNN with HyPar: Global Average Pooling convolutional neural network with novel NNLU activation function and HYBRID parallelism”. In: *Frontiers in Computational Neuroscience* 16 (2022). DOI: 10.3389/fncom.2022.1004988.
- [R89] Abien Fred Agarap. “Deep learning using rectified linear units (relu)”. In: *arXiv preprint arXiv:1803.08375* (2018).
- [R90] I. Kouretas and V. Paliouras. “Simplified Hardware Implementation of the Softmax Activation Function”. In: *2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST)*. 2019, pp. 1–4. DOI: 10.1109/MOCAST.2019.8741677.
- [R91] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].
- [R92] Elliott Gordon-Rodriguez, Gabriel Loaiza-Ganem, Geoff Pleiss, and John P. Cunningham. *Uses and Abuses of the Cross-Entropy Loss: Case Studies in Modern Deep Learning*. 2020. arXiv: 2011.05231 [stat.ML].
- [R93] Anna Czmil, Sylwester Czmil, and Damian Mazur. “A Method to Detect Type 1 Diabetes Based on Physical Activity Measurements Using a Mobile Device”. In: *Applied Sciences* 9.12 (2019). DOI: 10.3390/app9122555.
- [R94] Paul T Fuglestad, Meg Bruening, Dan Graham, Marla E Eisenberg, and Dianne R Neumark-Sztainer. “The Associations of Eating-related Attitudinal Balance with Psychological Well-being and Eating Behaviors”. In: *J Soc Clin Psychol* 32 (2013), pp. 1040–1060.
- [R95] Wesley R Barnhart, Lindsay Hamilton, Amy K Jordan, Mercedes Pratt, and Dara R Musher-Eizenman. “The interaction of negative psychological well-bein and picky eating in relation to disordered eating in undergraduate students”. In: *Eat Behav* 40 (Feb. 2021), p. 101476.

- [R96] Andrew P. Smith. “Eating, Drinking, and Well-Being”. In: *Handbook of Eating and Drinking: Interdisciplinary Perspectives*. Ed. by Herbert L. Meiselman. Springer International Publishing, 2020, pp. 765–785. DOI: 10.1007/978-3-030-14504-0_174.
- [R97] David G. Schlundt and Crystal Bell. “Behavioral assessment of eating patterns and blood glucose in diabetes using the Self-Monitoring Analysis System”. In: *Behavior Research Methods, Instruments, & Computers* 19 (Mar. 1987), pp. 215–223. DOI: 10.3758/BF03203788.
- [R98] Anthony P Winston. “Eating Disorders and Diabetes”. In: *Curr Diab Rep* 20 (June 2020), p. 32.
- [R99] Raul I. Ramos-Garcia, Eric R. Muth, John N. Gowdy, and Adam W. Hoover. “Improving the Recognition of Eating Gestures Using Intergesture Sequential Dependencies”. In: *IEEE Journal of Biomedical and Health Informatics* 19 (2015), pp. 825–831. DOI: 10.1109/JBHI.2014.2329137.
- [R100] John P Corbett, Liana Hsu, Sue A Brown, Laura Kollar, Katelijn Vleugels, Bruce Buckingham, et al. “Smartwatch gesture-based meal reminders improve glycaemic control”. In: *Diabetes Obes Metab* 24 (May 2022), pp. 1667–1670.
- [R101] Dario Ortega Anderrez, Ahmad Lotfi, and Amir Pourabdollah. “Eating and drinking gesture spotting and recognition using a novel adaptive segmentation technique and a gesture discrepancy measure”. In: *Expert Systems with Applications* 140 (2020), p. 112888. DOI: 10.1016/j.eswa.2019.112888.
- [R102] Dagmar Schuller and Björn Schuller. “The Challenge of Automatic Eating Behaviour Analysis and Tracking”. In: Jan. 2020, pp. 187–204. DOI: 10.1007/978-3-030-30817-9_8.
- [R103] Dario Ortega Anderrez, Ahmad Lotfi, and Amir Pourabdollah. “A deep learning based wearable system for food and drink intake recognition”. In: *Journal of Ambient Intelligence and Humanized Computing* 12 (Oct. 2021), pp. 9435–9447. DOI: 10.1007/s12652-020-02684-7.
- [R104] Dario Ortega Anderrez, Ahmad Lotfi, and Amir Pourabdollah. “Temporal convolution neural network for food and drink intake recognition”. In: *Proceedings of the 12th ACM International Conference on Pervasive Technologies Related to Assistive Environments*. 2019, pp. 580–586. DOI: 10.1145/3316782.3322784.
- [R105] Philipp V. Rouast, Hamid Heydarian, Marc T. P. Adam, and Megan E. Rollo. “OREBA: A Dataset for Objectively Recognizing Eating Behavior and Associated Intake”. In: *IEEE Access* 8 (2020), pp. 181955–181963. DOI: 10.1109/access.2020.3026965.
- [R106] George Sung, Kanstantsin Sokal, Esha Uboweja, Valentin Bazarevsky, Jonathan Bacash, Eduard Gabriel Bazavan, et al. *On-device Real-time Hand Gesture Recognition*. 2021. arXiv: 2111.00038 [cs.CV].
- [R107] Lingchen Chen, Feng Wang, Hui Deng, and Kaifan Ji. “A Survey on Hand Gesture Recognition”. In: *2013 International Conference on Computer Sciences and Applications*. 2013, pp. 313–316. DOI: 10.1109/CSA.2013.79.
- [R108] Jonas Bokstaller and Costanza Maria Improta. *Dynamic Gesture Recognition*. 2021. arXiv: 2109.09396 [cs.CV].
- [R109] Sushmita Mitra and Tinku Acharya. “Gesture Recognition: A Survey”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37 (2007), pp. 311–324. DOI: 10.1109/TSMCC.2007.893280.
- [R110] Min Zheng, Baohua Ni, and Samantha Kleinberg. “Automated meal detection from continuous glucose monitor data through simulation and explanation”. en. In: *J Am Med Inform Assoc* 26.12 (Dec. 2019), pp. 1592–1599.

- [R111] T. Kumar, Elise S iland,  yvind Stavdahl, and Anders Fougner. “Pilot Study of Early Meal Onset Detection from Abdominal Sounds”. In: Nov. 2019.
- [R112] Invensense. *MPU-6000 and MPU-6050 Product Specification Revision 3.4*. <https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Register-Map1.pdf>. Accessed: 2025-03-27. 2013.
- [R113] Panoramic Digital Health. “Panoramic Digital Wearable Health Sensors”. In: (2021). URL: <https://www.panoramicdigitalhealth.com/what-we-do.html>.
- [R114] Attila Zolt n Jenei, G bor Kiss, Mikl s G briel Tulics, and D vid Sztah . “Separation of Several Illnesses Using Correlation Structures with Convolutional Neural Networks”. In: *Acta Polytech. Hung* 18 (2021), pp. 47–66.
- [R115] Iqbal H. Sarker. “Machine Learning: Algorithms, Real-World Applications and Research Directions”. In: *SN Computer Science* 2 (Mar. 2021), p. 160. DOI: 10.1007/s42979-021-00592-x.
- [R116] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. “A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures”. In: *Neural Computation* 31 (July 2019), pp. 1235–1270. DOI: 10.1162/neco_a_01199.
- [R117] Jason Brownlee. *Difference between return sequences and return states for LSTMs in Keras*. 2019.
- [R118] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15 (2014), pp. 1929–1958.
- [R119] Boxue Zhang, Qi Zhao, Wenquan Feng, and Shuchang Lyu. “AlphaMEX: A smarter global pooling method for convolutional neural networks”. In: *Neurocomputing* 321 (2018), pp. 36–48. DOI: 10.1016/j.neucom.2018.07.079.
- [R120] Supratik Chatterjee and Arvind Keprate. “Predicting Remaining Fatigue Life of Topside Piping Using Deep Learning”. In: May 2021. DOI: 10.1109/ICAPAI49758.2021.9462055.
- [R121] Diederik Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *International Conference on Learning Representations* (Dec. 2014).
- [R122] Gabriela Da Silva Xavier. “The Cells of the Islets of Langerhans”. en. In: *J Clin Med* 7.3 (Mar. 2018).
- [R123] Ioannis Ogrotis, Theocharis Koufakis, and Kalliopi Kotsa. “Changes in the Global Epidemiology of Type 1 Diabetes in an Evolving Landscape of Environmental Factors: Causes, Challenges, and Opportunities”. en. In: *Medicina (Kaunas)* 59.4 (Mar. 2023).
- [R124] Alexia G Abela and Stephen Fava. “Why is the Incidence of Type 1 Diabetes Increasing?” en. In: *Curr Diabetes Rev* 17.8 (2021), e030521193110.
- [R125] Andrej Jane , Cristian Guja, Asimina Mitrakou, Nebojsa Lalic, Tsvetalina Tankova, Leszek Czupryniak, et al. “Insulin Therapy in Adults with Type 1 Diabetes Mellitus: a Narrative Review”. en. In: *Diabetes Ther* 11.2 (Jan. 2020), pp. 387–409.
- [R126] Carlos E Mendez and Guillermo Umpierrez. “Management of the hospitalized patient with type 1 diabetes mellitus”. en. In: *Hosp Pract (1995)* 41.3 (Aug. 2013), pp. 89–100.
- [R127] Iv n Contreras, Silvia Oviedo, Martina Vettoretti, Roberto Visentin, and Josep Veh . “Personalized blood glucose prediction: A hybrid approach using grammatical evolution and physiological models”. en. In: *PLoS One* 12.11 (Nov. 2017), e0187754.
- [R128] Axel C M hlbacher, Andrew Sadler, and Christin Juhnke. “Personalized diabetes management: what do patients with diabetes mellitus prefer? A discrete choice experiment”. en. In: *Eur J Health Econ* 22.3 (Feb. 2021), pp. 425–443.

- [R129] Emmanouil G Spanakis, Franco Chiarugi, Angelina Kouroubali, Stephan Spat, Peter Beck, Stefan Asanin, et al. “Diabetes management using modern information and communication technologies and new care models”. en. In: *Interact J Med Res* 1.2 (Oct. 2012), e8.
- [R130] Marta Bassi, Daniele Franzone, Francesca Dufour, Marina Francesca Strati, Marta Scalas, Giacomo Tantari, et al. “Automated Insulin Delivery (AID) Systems: Use and Efficacy in Children and Adults with Type 1 Diabetes and Other Forms of Diabetes in Europe in Early 2023”. en. In: *Life (Basel)* 13.3 (Mar. 2023).
- [R131] Jennifer L. Sherr, Lutz Heinemann, G. Alexander Fleming, Richard M. Bergenstal, Daniela Bruttomesso, Hélène Hanaire, et al. “Automated Insulin Delivery: Benefits, Challenges, and Recommendations. A Consensus Report of the Joint Diabetes Technology Working Group of the European Association for the Study of Diabetes and the American Diabetes Association”. In: *Diabetes Care* 45.12 (Oct. 2022), pp. 3058–3074. ISSN: 0149-5992. DOI: 10.2337/dci22-0018. eprint: <https://diabetesjournals.org/care/article-pdf/45/12/3058/692853/dci220018.pdf>. URL: <https://doi.org/10.2337/dci22-0018>.
- [R132] Claudio Cobelli, Eric Renard, and Boris Kovatchev. “Artificial pancreas: past, present, future”. en. In: *Diabetes* 60.11 (Nov. 2011), pp. 2672–2682.
- [R133] Sun Joon Moon, Inha Jung, and Cheol-Young Park. “Current Advances of Artificial Pancreas Systems: A Comprehensive Review of the Clinical Evidence”. en. In: *Diabetes Metab J* 45.6 (Nov. 2021), pp. 813–839.
- [R134] SA Brown, BP Kovatchev, D Raghinaru, and ”et al.” “iDCL Trial in the New England Journal of Medicine”. In: *The New England Journal of Medicine* 381.18 (2019), pp. 1707–1717. DOI: 10.1056/NEJMoa1907863.
- [R135] Roman Hovorka. “Closed-loop insulin delivery: from bench to clinical practice”. In: *Nature Reviews Endocrinology* 7.7 (2011), pp. 385–395.
- [R136] Kamuran Turksoy and Ali Cinar. “Adaptive control of artificial pancreas systems - a review”. en. In: *J Healthc Eng* 5.1 (2014), pp. 1–22.
- [R137] Griselda Quiroz. “The evolution of control algorithms in artificial pancreas: A historical perspective”. In: *Annual Reviews in Control* 48 (2019), pp. 222–232. ISSN: 1367-5788. DOI: <https://doi.org/10.1016/j.arcontrol.2019.07.004>. URL: <https://www.sciencedirect.com/science/article/pii/S1367578819300148>.
- [R138] John Bagterp Jørgensen, Dimitri Boiroux, and Zeinab Mahmoudi. “An artificial pancreas based on simple control algorithms and physiological insight”. In: *IFAC-PapersOnLine* 52.1 (2019). 12th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems DYCOPS 2019, pp. 1018–1023. ISSN: 2405-8963. DOI: <https://doi.org/10.1016/j.ifacol.2019.06.196>. URL: <https://www.sciencedirect.com/science/article/pii/S2405896319302848>.
- [R139] Yazdan Batmani, Shadi Khodakaramzadeh, and Parham Moradi. “Automatic Artificial Pancreas Systems Using an Intelligent Multiple-Model PID Strategy”. en. In: *IEEE J Biomed Health Inform* 26.4 (Apr. 2022), pp. 1708–1717.
- [R140] Esther Matamoros-Alcivar, Tanya Ascencio-Lino, Rigoberto Fonseca, Gandhi Villalba-Meneses, Andrés Tirado-Espín, Lorena Barona, et al. “Implementation of MPC and PID Control Algorithms to the Artificial Pancreas for Diabetes Mellitus Type 1”. In: *2021 IEEE International Conference on Machine Learning and Applied Network Technologies (ICMLANT)*. 2021, pp. 1–6. DOI: 10.1109/ICMLANT53170.2021.9690529.

- [R141] Lauren M Huyett, Eyal Dassau, Howard C Zisser, and Francis J Doyle 3rd. “Design and Evaluation of a Robust PID Controller for a Fully Implantable Artificial Pancreas”. en. In: *Ind Eng Chem Res* 54.42 (June 2015), pp. 10311–10321.
- [R142] Su Lim Kang, Yoo Na Hwang, Ji Yean Kwon, and Sung Min Kim. “Effectiveness and safety of a model predictive control (MPC) algorithm for an artificial pancreas system in outpatients with type 1 diabetes (T1D): systematic review and meta-analysis”. en. In: *Diabetol Metab Syndr* 14.1 (Dec. 2022), p. 187.
- [R143] Richard Mauseth, Irl B Hirsch, Jennifer Bollyky, Robert Kircher, Don Matheson, Srinath Sanda, et al. “Use of a “fuzzy logic” controller in a closed-loop artificial pancreas”. en. In: *Diabetes Technol Ther* 15.8 (July 2013), pp. 628–633.
- [R144] Atlas, Eran, Revital Nimri, Shahar Miller, Eli A Grunberg, and Moshe Phillip. “MD-logic artificial pancreas system: a pilot study in adults with type 1 diabetes”. en. In: *Diabetes Care* 33.5 (Feb. 2010), pp. 1072–1076.
- [R145] Seunghyun Lee, Jiwon Kim, Sung Woon Park, Sang-Man Jin, and Sung-Min Park. “Toward a Fully Automated Artificial Pancreas System Using a Bioinspired Reinforcement Learning Design: In Silico Validation”. en. In: *IEEE J Biomed Health Inform* 25.2 (Feb. 2021), pp. 536–546.
- [R146] João Lucas Correia Barbosa de Farias and Wallace Moreira Bessa. “Intelligent Control with Artificial Neural Networks for Automated Insulin Delivery Systems”. en. In: *Bioengineering (Basel)* 9.11 (Nov. 2022).
- [R147] Phuwadol Viroonluecha, Esteban Egea-Lopez, and Jose Santa. “Evaluation of blood glucose level control in type 1 diabetic patients using deep reinforcement learning”. en. In: *PLOS ONE* 17.9 (Sept. 2022). Publisher: Public Library of Science, e0274608. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0274608. URL: <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0274608>.
- [R148] Miguel Tejedor, Ashenafi Zebene Woldaregay, and Fred Godtliebsen. “Reinforcement learning application in diabetes blood glucose control: A systematic review”. en. In: *Artificial Intelligence in Medicine* 104 (Apr. 2020), p. 101836. ISSN: 0933-3657. DOI: 10.1016/j.artmed.2020.101836.
- [R149] Jelena Tašić, Márta Takács, and Levente Kovács. “Control Engineering Methods for Blood Glucose Levels Regulation”. In: *Acta Polytechnica Hungarica* 19 (Aug. 2022), pp. 127–152. DOI: 10.12700/APH.19.7.2022.7.7.
- [R150] Julien Perolat, Bart De Vylder, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer, et al. “Mastering the game of Stratego with model-free multiagent reinforcement learning”. In: *Science* 378.6623 (Dec. 2022). Publisher: American Association for the Advancement of Science, pp. 990–996. DOI: 10.1126/science.add4679. URL: <https://www.science.org/doi/abs/10.1126/science.add4679>.
- [R151] Yongkui Liu, He Xu, Ding Liu, and Lihui Wang. “A digital twin-based sim-to-real transfer for deep reinforcement learning-enabled industrial robot grasping”. en. In: *Robotics and Computer-Integrated Manufacturing* 78 (Dec. 2022), p. 102365. ISSN: 0736-5845. DOI: 10.1016/j.rcim.2022.102365.
- [R152] Siqi Liu, Kay Choong See, Kee Yuan Ngiam, Leo Anthony Celi, Xingzhi Sun, and Mengling Feng. “Reinforcement Learning for Clinical Decision Support in Critical Care: Comprehensive Review”. EN. In: *Journal of Medical Internet Research* 22.7 (July 2020), e18477. DOI: 10.2196/18477. URL: <https://www.jmir.org/2020/7/e18477>.

- [R153] Iqra Shafeeq Mughal, Luca Patanè, and Riccardo Caponetto. “A comprehensive review of models and nonlinear control strategies for blood glucose regulation in artificial pancreas”. In: *Annual Reviews in Control* 57 (2024), p. 100937. ISSN: 1367-5788. DOI: <https://doi.org/10.1016/j.arcontrol.2024.100937>. URL: <https://www.sciencedirect.com/science/article/pii/S1367578824000063>.
- [R154] Sami S Kanderian, Stuart A Weinzimer, and Garry M Steil. “The identifiable virtual patient model: comparison of simulation and clinical closed-loop study results”. en. In: *J Diabetes Sci Technol* 6.2 (Mar. 2012), pp. 371–379.
- [R155] Martina Vettoretti, Cristina Battocchio, Giovanni Sparacino, and Andrea Facchinetti. “Development of an Error Model for a Factory-Calibrated Continuous Glucose Monitoring Sensor with 10-Day Lifetime”. In: *Sensors* 19.23 (2019). ISSN: 1424-8220. DOI: 10.3390/s19235320. URL: <https://www.mdpi.com/1424-8220/19/23/5320>.
- [R156] L. M. Huyett, E. Dassau, H. C. Zisser, and F. J. Doyle. “Glucose Sensor Dynamics and the Artificial Pancreas: The Impact of Lag on Sensor Measurement and Controller Performance”. In: *IEEE Control Systems Magazine* 38.1 (2018), pp. 30–46. DOI: 10.1109/MCS.2017.2766322.
- [R157] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. en. Ed. by Francis Bach. 2nd ed. Adaptive Computation and Machine Learning series. Cambridge, MA, USA: A Bradford Book, Nov. 2018. ISBN: 978-0-262-03924-6.
- [R158] Vijay Konda and John Tsitsiklis. “Actor-Critic Algorithms”. In: *Society for Industrial and Applied Mathematics* 42 (Apr. 2001).
- [R159] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. *Proximal Policy Optimization Algorithms*. 2017. DOI: 10.48550/ARXIV.1707.06347. URL: <https://arxiv.org/abs/1707.06347>.
- [R160] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. “Stable-Baselines3: Reliable Reinforcement Learning Implementations”. In: *Journal of Machine Learning Research* 22.268 (2021), pp. 1–8. URL: <http://jmlr.org/papers/v22/20-1364.html>.
- [R161] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, et al. “Openai gym”. In: *arXiv preprint arXiv:1606.01540* (2016).
- [R162] Arsenii Kuznetsov, Pavel Shvechikov, Alexander Grishin, and Dmitry Vetrov. *Controlling Overestimation Bias with Truncated Mixture of Continuous Distributional Quantile Critics*. 2020. arXiv: 2005.04269 [cs.LG]. URL: <https://arxiv.org/abs/2005.04269>.
- [R163] Zihao Wang, Zhiqiang Xie, Enmei Tu, Alex Zhong, Yingying Liu, Jichang Ding, et al. “Reinforcement Learning-Based Insulin Injection Time And Dosages Optimization”. In: *2021 International Joint Conference on Neural Networks (IJCNN)*. ISSN: 2161-4407. July 2021, pp. 1–8. DOI: 10.1109/IJCNN52387.2021.9533957.
- [R164] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. “Proximal policy optimization algorithms”. In: *arXiv preprint arXiv:1707.06347* (2017).
- [R165] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. “Soft actor-critic algorithms and applications”. In: *arXiv preprint arXiv:1812.05905* (2018).
- [R166] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, et al. “Continuous control with deep reinforcement learning”. In: *arXiv preprint arXiv:1509.02971* (2015).
- [R167] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, et al. “Human-level control through deep reinforcement learning”. In: *Nature* 518.7540 (2015), pp. 529–533.

- [R168] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, et al. “Asynchronous methods for deep reinforcement learning”. In: *International conference on machine learning* (2016), pp. 1928–1937.
- [R169] Scott Fujimoto, Herke van Hoof, and David Meger. “Addressing function approximation error in actor-critic methods”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1582–1591.
- [R170] R. Fry and S. McManus. “Smooth bump functions and the geometry of banach spaces: A brief survey”. In: *Expositiones Mathematicae* 20.2 (2002), pp. 143–183. ISSN: 0723-0869. DOI: [https://doi.org/10.1016/S0723-0869\(02\)80017-2](https://doi.org/10.1016/S0723-0869(02)80017-2). URL: <https://www.sciencedirect.com/science/article/pii/S0723086902800172>.
- [R171] Abhishek Singh, Aparna Rawat, and Nikhila Raghuthaman. “Mexican Hat Wavelet Transform and Its Applications”. In: *Methods of Mathematical Modelling and Computation for Complex Systems*. Ed. by Jagdev Singh, Hemen Dutta, Devendra Kumar, Dumitru Baleanu, and Jordan Hristov. Cham: Springer International Publishing, 2022, pp. 299–317. ISBN: 978-3-030-77169-0. DOI: 10.1007/978-3-030-77169-0_12. URL: https://doi.org/10.1007/978-3-030-77169-0_12.
- [R172] N. Gordillo, E. Montseny, and P. Sobrevilla. “State of the art survey on MRI brain tumor segmentation”. In: *Magn. Reson. Imaging* 31 (2013), pp. 1426–1438.
- [R173] G. Mohan and M.M. Subashini. “MRI based medical image analysis: Survey on brain tumor grade classification”. In: *Biomed. Sign. Proc. Contr.* 39 (2018), pp. 139–161.
- [R174] L. Wang, D. Nie, G. N Li, É. Puybureau, J. Dolz, Q. Zhang, et al. “Benchmark on automatic 6-month-old infant brain segmentation algorithms: the iSeg-2017 challenge”. In: *IEEE Trans. Med. Imag.* 38 (2019), pp. 2219–2230.
- [R175] Yue Sun, Kun Gao, Zhengwang Wu, Guannan Li, Xiaopeng Zong, Zhihao Lei, et al. “Multi-Site Infant Brain Segmentation Algorithms: The iSeg-2019 Challenge”. In: *IEEE Transactions on Medical Imaging* 40.5 (2021), pp. 1363–1376.
- [R176] Valeria Diaz and Guillermo Rodríguez. “Machine learning for detection of cognitive impairment”. In: *Acta Polytech. Hung.* 19.5 (2022), pp. 196–213.
- [R177] P. Moeskops, M. J. N. L. Benders, S. M. Chiță, K. J. Kersbergen, F. Groenendaal, L. S. de Vries, et al. “Automatic segmentation of MR brain images of preterm infants using supervised classification”. In: *NeuroImage* 118 (2015), pp. 628–641.
- [R178] Z. Kapás, L. Lefkovits, and L. Szilágyi. “Automatic detection and segmentation of brain tumor using random forest approach”. In: *Modeling Decisions for Artificial Intelligence, LNCS 9880*. 2016, pp. 301–312.
- [R179] L. Wang, F. Shi, G. Li, W. L. Lin, and D. G. Gilmore J. H. and Shen. “Integration of sparse multi-modality representation and geometrical constraint for isointense infant brain segmentation”. In: *International Conference on Medical Image Computation and Computer Assisted Interventions, LNCS 8149*. 2013, pp. 703–710.
- [R180] M. Ismail, M. Mostapha, A. Soliman, M. Nitzken, F. Khalifa, A. Elnakib, et al. “Segmentation of infant brain MR images on adaptive shape prior and higher-order MGRF”. In: *International Conference on Information Processing*. 2015, pp. 4327–4331.
- [R181] A. Alansary, A. Soliman, M. Nitzken, F. Khalifa, A. Elnakib, M. Mostapha, et al. “An integrated geometrical and stochastic approach for accurate infant brain extraction”. In: *2014 IEEE International Conference on Image Processing (ICIP)*. 2014, pp. 3542–3546. DOI: 10.1109/ICIP.2014.7025719.
- [R182] P. Dong, L. Wang, W. L. Lin, D. G. Shen, and G. R. Wu. “Scalable joint segmentation and registration framework for infant brain images”. In: *Neurocomput.* 229 (2017), pp. 54–62.

- [R183] L. Szilágyi, S. M. Szilágyi, B. Benyó, and Z. Benyó. “Intensity inhomogeneity compensation and segmentation of MR brain images using hybrid *c*-means clustering models”. In: *Biomed. Sign. Proc. Contr.* 6 (2011), pp. 3–12.
- [R184] L. Szilágyi, L. Lefkovits, and B. Benyó. “Automatic brain tumor segmentation in multispectral MRI volumes using a fuzzy *c*-means cascade algorithm”. In: *Proc. 12th International Conference on Fuzzy Systems and Knowledge Discovery*. 2015, pp. 285–291.
- [R185] Jose Bernal, Kaisar Kushibar, Mariano Cabezas, Sergi Valverde, Arnau Oliver, and Xavier Lladó. “Quantitative Analysis of Patch-Based Fully Convolutional Neural Networks for Tissue Segmentation on Brain Magnetic Resonance Imaging”. In: *IEEE Access* 7 (2019), pp. 89986–90002. DOI: 10.1109/ACCESS.2019.2926697.
- [R186] M. Mostapha and M. Styner. “Role of deep learning in infant brain MRI analysis”. In: *Magnet. Res. Imag.* 64 (2019), pp. 171–189.
- [R187] Jose Dolz, Christian Desrosiers, Li Wang, Jing Yuan, Dinggang Shen, and Ismail Ben Ayed. “Deep CNN ensembles and suggestive annotations for infant brain MRI segmentation”. In: *Computerized Medical Imaging and Graphics* 79 (2020), p. 101660. ISSN: 0895-6111. DOI: <https://doi.org/10.1016/j.compmedimag.2019.101660>. URL: <https://www.sciencedirect.com/science/article/pii/S0895611119300771>.
- [R188] Q. Zhang, L. Wang, X. P. Zong, W. L. Lin, G. Li, and D. G. Shen. “FRNET: flattened residual network for infant MRI skull stripping”. In: *International Symposium on Biomedical Imaging*. 2019, pp. 999–1002.
- [R189] S. Qamar, H. Jin, R. Zheng, P. Ahmad, and P. Usama. “A variant form of 3D-UNet for infant brain segmentation”. In: *Future Gener. Comp. Syst.* 108 (2020), pp. 613–623.
- [R190] U. Vovk, F. Pernuš, and B. Likar. “A review of methods for correction of intensity inhomogeneity in MRI”. In: *IEEE Trans. Med. Imag.* 26 (2007), pp. 405–421.
- [R191] L. Szilágyi, S. M. Szilágyi, and B. Benyó. “Efficient inhomogeneity compensation using fuzzy *c*-means clustering models”. In: *Comput. Meth. Progr. Biomed.* 108 (2012), pp. 80–89.
- [R192] L. G. Nyúl, J. K. Udupa, and X. Zhang. “New variants of a method of MRI scale standardization”. In: *IEEE Trans. Med. Imag.* 19 (2000), pp. 143–150.
- [R193] A. Köble, A. Györfi, S. Csaholczi, B. Surányi, L. Dénes-Fazakas, L. Kovács, et al. “Identifying the most suitable histogram normalization technique for machine learning based segmentation of multispectral brain MRI data”. In: *Proc. IEEE AFRICON*. 2021, pp. 71–76.
- [R194] O. Ronneberger, P. Fischer, and T. Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *Proc. Medical Image Computation and Computer-Assisted Interventions (MICCAI), LNCS vol. 9351*. 2015, pp. 234–241.
- [R195] Diederik P. Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).
- [R196] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. “Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)”. In: *ICLR 2016*. Nov. 2015.
- [R197] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. *A Closer Look at Spatiotemporal Convolutions for Action Recognition*. 2018. arXiv: 1711.11248 [cs.CV]. URL: <https://arxiv.org/abs/1711.11248>.
- [R198] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. *Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)*. 2016. arXiv: 1511.07289 [cs.LG].

- [R199] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].
- [R200] B. Surányi, L. Kovács, and L. Szilágyi. “Segmentation of brain tissues from infant MRI records using machine learning techniques”. In: *Proc. 19th IEEE World Symposium on Applied Machine Intelligence and Informatics (SAMi)*. 2021, pp. 455–460.
- [R201] Tan Nguyen, Binh-Son Hua, and Ngan Le. “3D-UCaps: 3D Capsules Unet for Volumetric Image Segmentation”. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*. Ed. by Marleen de Bruijne, Philippe C. Cattin, Stéphane Cotin, Nicolas Padoy, Stefanie Speidel, Yefeng Zheng, et al. Cham: Springer International Publishing, 2021, pp. 548–558. ISBN: 978-3-030-87193-2.
- [R202] Minh Tran, Loi Ly, Binh-Son Hua, and Ngan Le. “SS-3DCAPSNET: Self-Supervised 3d Capsule Networks for Medical Segmentation on Less Labeled Data”. In: *2022 IEEE 19th International Symposium on Biomedical Imaging (ISBI)*. 2022, pp. 1–5. DOI: 10.1109/ISBI52829.2022.9761627.
- [R203] Valery L Feigin, Emma Nichols, Tahiya Alam, Marlena S Bannick, Ettore Beghi, Nat-acha Blake, et al. “Global, regional, and national burden of neurological disorders, 1990–2016: a systematic analysis for the Global Burden of Disease Study 2016”. In: *The Lancet Neurology* 18.5 (2019), pp. 459–480.
- [R204] Bjoern H. Menze, Andras Jakab, Stefan Bauer, Jayashree Kalpathy-Cramer, Keyvan Farahani, Justin Kirby, et al. “The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)”. In: *IEEE Transactions on Medical Imaging* 34.10 (2015), pp. 1993–2024. DOI: 10.1109/TMI.2014.2377694.
- [R205] Spyridon Bakas, Mauricio Reyes, Andras Jakab, Stefan Bauer, Markus Rempfler, Alessandro Crimi, et al. *Identifying the Best Machine Learning Algorithms for Brain Tumor Segmentation, Progression Assessment, and Overall Survival Prediction in the BRATS Challenge*. 2019. arXiv: 1811.02629 [cs.CV].
- [R206] Ágnes Györfi, László Szilágyi, and Levente Kovács. “A Fully Automatic Procedure for Brain Tumor Segmentation from Multi-Spectral MRI Records Using Ensemble Learning and Atlas-Based Data Enhancement”. In: *Applied Sciences* 11.2 (2021). ISSN: 2076-3417. DOI: 10.3390/app11020564. URL: <https://www.mdpi.com/2076-3417/11/2/564>.
- [R207] Peter Macsik, Jarmila Pavlovicova, Jozef Goga, and Slavomir Kajan. “Local binary CNN for diabetic retinopathy classification on fundus images”. In: *Acta Polytech. Hung* 19.7 (2022), pp. 27–45.
- [R208] Patrik Szepesi and László Szilágyi. “Detection of pneumonia using convolutional neural networks and deep learning”. In: *Biocybernetics and Biomedical Engineering* 42.3 (2022), pp. 1012–1022. ISSN: 0208-5216. DOI: <https://doi.org/10.1016/j.bbe.2022.08.001>. URL: <https://www.sciencedirect.com/science/article/pii/S0208521622000742>.
- [R209] G. Neelima, Dhanunjaya Rao Chigurukota, Balajee Maram, and B. Girirajan. “Optimal DeepMRSeg based tumor segmentation with GAN for brain tumor classification”. In: *Biomedical Signal Processing and Control* 74 (2022), p. 103537. ISSN: 1746-8094. DOI: <https://doi.org/10.1016/j.bspc.2022.103537>. URL: <https://www.sciencedirect.com/science/article/pii/S1746809422000593>.
- [R210] Pendela Kanchanamala, Revathi K.G., and M. Belsam Jeba Ananth. “Optimization-enabled hybrid deep learning for brain tumor detection and classification from MRI”. In: *Biomedical Signal Processing and Control* 84 (2023), p. 104955. ISSN: 1746-8094. DOI: <https://doi.org/10.1016/j.bspc.2023.104955>. URL: <https://www.sciencedirect.com/science/article/pii/S1746809423003889>.

- [R211] S K Rajeev, M. Pallikonda Rajasekaran, G. Vishnuvarthanan, and T. Arunprasath. “A biologically-inspired hybrid deep learning approach for brain tumor classification from magnetic resonance imaging using improved gabor wavelet transform and Elmann-BiLSTM network”. In: *Biomedical Signal Processing and Control* 78 (2022), p. 103949. ISSN: 1746-8094. DOI: <https://doi.org/10.1016/j.bspc.2022.103949>. URL: <https://www.sciencedirect.com/science/article/pii/S1746809422004487>.
- [R212] Lalita Mishra and Shekhar Verma. “Graph attention autoencoder inspired CNN based brain tumor classification using MRI”. In: *Neurocomputing* 503 (2022), pp. 236–247. ISSN: 0925-2312. DOI: <https://doi.org/10.1016/j.neucom.2022.06.107>. URL: <https://www.sciencedirect.com/science/article/pii/S0925231222008384>.
- [R213] K. Rasool Reddy and Ravindra Dhuli. “Segmentation and classification of brain tumors from MRI images based on adaptive mechanisms and ELDP feature descriptor”. In: *Biomedical Signal Processing and Control* 76 (2022), p. 103704. ISSN: 1746-8094. DOI: <https://doi.org/10.1016/j.bspc.2022.103704>. URL: <https://www.sciencedirect.com/science/article/pii/S1746809422002269>.
- [R214] Hossein Mehnatkesh, Seyed Mohammad Jafar Jalali, Abbas Khosravi, and Saeid Naha-vandi. “An intelligent driven deep residual learning framework for brain tumor classification using MRI images”. In: *Expert Systems with Applications* 213 (2023), p. 119087. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2022.119087>. URL: <https://www.sciencedirect.com/science/article/pii/S0957417422021054>.
- [R215] Ramdas Vankdothu and Mohd Abdul Hameed. “Brain tumor MRI images identification and classification based on the recurrent convolutional neural network”. In: *Measurement: Sensors* 24 (2022), p. 100412. ISSN: 2665-9174. DOI: <https://doi.org/10.1016/j.measen.2022.100412>. URL: <https://www.sciencedirect.com/science/article/pii/S2665917422000460>.
- [R216] Takowa Rahman and Md Saiful Islam. “MRI brain tumor detection and classification using parallel deep convolutional neural networks”. In: *Measurement: Sensors* 26 (2023), p. 100694. ISSN: 2665-9174. DOI: <https://doi.org/10.1016/j.measen.2023.100694>. URL: <https://www.sciencedirect.com/science/article/pii/S2665917423000302>.
- [R217] B. Venkateswarlu Isunuri and Jagadeesh Kakarla. “EfficientNet and multi-path convolution with multi-head attention network for brain tumor grade classification”. In: *Computers and Electrical Engineering* 108 (2023), p. 108700. ISSN: 0045-7906. DOI: <https://doi.org/10.1016/j.compeleceng.2023.108700>. URL: <https://www.sciencedirect.com/science/article/pii/S0045790623001246>.
- [R218] Jun Cheng, Wei Huang, Shuangliang Cao, Ru Yang, Wei Yang, Zhaoqiang Yun, et al. “Enhanced Performance of Brain Tumor Classification via Tumor Region Augmentation and Partition”. en. In: *PLoS One* 10.10 (Oct. 2015), e0140381.
- [R219] Jun Cheng. “brain tumor dataset”. In: (Apr. 2017). DOI: 10.6084/m9.figshare.1512427.v5. URL: https://figshare.com/articles/dataset/brain_tumor_dataset/1512427.
- [R220] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].
- [R221] Keiron O’Shea and Ryan Nash. *An Introduction to Convolutional Neural Networks*. 2015. arXiv: 1511.08458 [cs.NE]. URL: <https://arxiv.org/abs/1511.08458>.
- [R222] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. “Dropout: a simple way to prevent neural networks from overfitting”. In: 15.1 (Jan. 2014), pp. 1929–1958. ISSN: 1532-4435.

- [R223] Afia Zafar, Muhammad Aamir, Nazri Mohd Nawi, Ali Arshad, Saman Riaz, Abdulrahman Alruban, et al. “A Comparison of Pooling Methods for Convolutional Neural Networks”. In: *Applied Sciences* 12.17 (2022). ISSN: 2076-3417. DOI: 10.3390/app12178643. URL: <https://www.mdpi.com/2076-3417/12/17/8643>.
- [R224] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. *Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)*. 2016. arXiv: 1511.07289 [cs.LG]. URL: <https://arxiv.org/abs/1511.07289>.
- [R225] Abien Fred Agarap. *Deep Learning using Rectified Linear Units (ReLU)*. 2019. arXiv: 1803.08375 [cs.NE]. URL: <https://arxiv.org/abs/1803.08375>.

Own Publications Pertaining to Theses

- [J1] Lehel Dénes-Fazakas, Máté Siket, László Szilágyi, Levente Kovács, and György Eigner. “Detection of Physical Activity Using Machine Learning Methods Based on Continuous Blood Glucose Monitoring and Heart Rate Signals”. In: *Sensors* 22 (2022). DOI: 10.3390/s22218568.
- [J2] Lehel Dénes-Fazakas, Barbara Simon, Ádám Hartvég, Levente Kovács, Éva-Henrietta Dulf, László Szilágyi, et al. “Physical Activity Detection for Diabetes Mellitus Patients Using Recurrent Neural Networks”. In: *Sensors* 24.8 (2024). ISSN: 1424-8220. DOI: 10.3390/s24082412. URL: <https://www.mdpi.com/1424-8220/24/8/2412>.
- [J3] Lehel Dénes-Fazakas, Barbara Simon, Ádám Hartvég, László Szilágyi, Levente Kovács, Amir Mosavi, et al. “Personalized food consumption detection with deep learning and Inertial Measurement Unit sensor”. In: *Computers in Biology and Medicine* 182 (2024), p. 109167. ISSN: 0010-4825. DOI: <https://doi.org/10.1016/j.compbiomed.2024.109167>. URL: <https://www.sciencedirect.com/science/article/pii/S0010482524012526>.
- [J4] Lehel Dénes-Fazakas, László Szilágyi, Levente Kovács, Andrea De Gaetano, and György Eigner. “Reinforcement Learning: A Paradigm Shift in Personalized Blood Glucose Management for Diabetes”. In: *Biomedicines* 12.9 (2024). ISSN: 2227-9059. DOI: 10.3390/biomedicines12092143. URL: <https://www.mdpi.com/2227-9059/12/9/2143>.
- [J5] Lehel Dénes-Fazakas, György Eigner, Levente Kovács, and László Szilágyi. “Two U-net Architectures for Infant Brain Tissue Segmentation from Multi-Spectral MRI Data”. In: *IFAC-PapersOnLine* 56.2 (2023). 22nd IFAC World Congress, pp. 5637–5642. ISSN: 2405-8963. DOI: <https://doi.org/10.1016/j.ifacol.2023.10.479>. URL: <https://www.sciencedirect.com/science/article/pii/S2405896323008467>.
- [J6] Lehel Dénes-Fazakas, Levente Kovács, György Eigner, and László Szilágyi. “Enhancing Brain Tumor Diagnosis with L-Net: A Novel Deep Learning Approach for MRI Image Segmentation and Classification”. In: *Biomedicines* 12.10 (2024). ISSN: 2227-9059. DOI: 10.3390/biomedicines12102388. URL: <https://www.mdpi.com/2227-9059/12/10/2388>.
- [J7] Lehel Dénes-Fazakas, Levente Kovács, György Eigner, and László Szilágyi. “Enhanced U-Net for Infant Brain MRI Segmentation: A (2+1)D Convolutional Approach”. In: *Sensors* 25.5 (2025). ISSN: 1424-8220. DOI: 10.3390/s25051531. URL: <https://www.mdpi.com/1424-8220/25/5/1531>.
- [C1] Dénes-Fazakas Lehel, Máté Siket, László Szilágyi, György Eigner, and Levente Kovács. “Effect of Hyperparameters of Reinforcement Learning in Blood Glucose Control”. In: *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2023, pp. 1333–1340. DOI: 10.1109/SMC53992.2023.10393930.

- [C2] Dénes-Fazakas Lehel, Máté Siket, László Szilágyi, Gyorgy Eigner, and Levente Kovács. “Investigation of reward functions for controlling blood glucose level using reinforcement learning”. In: *2023 IEEE 17th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. 2023, pp. 000387–000392. DOI: 10.1109/SACI58269.2023.10158621.
- [C3] Lehel Dénes-Fazakas, Máté Siket, Gábor Kertész, László Szilágyi, Levente Kovács, and György Eigner. “Control of Type 1 Diabetes Mellitus using direct reinforcement learning based controller”. In: *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2022, pp. 1512–1517. DOI: 10.1109/SMC53654.2022.9945084.
- [C4] Marcell Szántó, Gergő Strasser, László Szász, Lehel Dénes-Fazakas, György Eigner, Gábor Kertész, et al. “Utilization of IMU-Based Gesture Recognition in the Treatment of Diabetes”. In: *2022 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*. 2022, pp. 1–5.
- [C5] Lehel Dénes-Fazakas, László Szilágyi, Jelena Tasic, Levente Kovács, and György Eigner. “Detection of physical activity using machine learning methods”. In: *2020 IEEE 20th International Symposium on Computational Intelligence and Informatics (CINTI)*. IEEE. 2020, pp. 167–172.
- [C6] Lehel Dénes-Fazakas, Győző Dénes Fazakas, György Eigner, Levente Kovács, and László Szilágyi. “Review of Reinforcement Learning-Based Control Algorithms in Artificial Pancreas Systems for Diabetes Mellitus Management”. In: *2024 IEEE 18th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. 2024, pp. 000565–000572. DOI: 10.1109/SACI60582.2024.10619866.
- [C7] Lehel Dénes-Fazakas, Levente Kovács, György Eigner, and László Szilágyi. “Brain Tumor Segmentation from Multi-Spectral MRI Records Using a U-Net Cascade Architecture”. In: *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2023, pp. 3003–3008. DOI: 10.1109/SMC53992.2023.10394588.
- [C8] Lehel Dénes-Fazakas, György Eigner, and László Szilágyi. “Segmentation of 6-month infant brain tissues from multi-spectral MRI records using a U-Net neural network architecture”. In: *2022 IEEE 10th Jubilee International Conference on Computational Cybernetics and Cyber-Medical Systems (ICCC)*. 2022, pp. 000077–000082. DOI: 10.1109/ICCC202255925.2022.9922800.

Publications not Pertaining to Theses

- [N1] György Eigner, Dénes-Fazekas L., László Szilágyi, Máté Siket, and Levente Kovács. “Physical Activity Detection Using Machine Intelligence. ATTD 2021 Invited Speaker Abstracts”. In: *DIABETES TECHNOLOGY AND THERAPEUTICS* 23 (2021), A-102-A-102. ISSN: 1520-9156. DOI: 10.1089/dia.2021.2525.abstracts.
- [N2] A. Köble, A. Györfi, S. Csaholczi, B. Surányi, L. Dénes-Fazakas, L. Kovács, et al. “Identifying the most suitable histogram normalization technique for machine learning based segmentation of multispectral brain MRI data”. In: *Proc. IEEE AFRICON*. 2021, pp. 71–76.
- [N3] Margit Antal and Lehel Denes-Fazakas. “User Verification Based on Mouse Dynamics: a Comparison of Public Data Sets”. In: *2019 IEEE 13th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. 2019, pp. 143–148. DOI: 10.1109/SACI46893.2019.9111596.
- [N4] Lehel Dénes-Fazakas, Eszter Kail, and Rita Fleiner. “Two-factor, continuous authentication framework for multi-site large enterprises”. In: *2020 IEEE 20th International Symposium on Computational Intelligence and Informatics (CINTI)*. 2020, pp. 173–178. DOI: 10.1109/CINTI51262.2020.9305817.
- [N5] Ágnes Györfi, Szabolcs Csaholczi, Ioan-Marius Lukáts-Pisak, Lehel Dénes-Fazakas, Andreea Köble, Olga Shvets, et al. “Effect of spectral resolution on the segmentation quality of magnetic resonance imaging data”. In: *2022 IEEE 26th International Conference on Intelligent Engineering Systems (INES)*. 2022, pp. 000053–000058. DOI: 10.1109/INES56734.2022.9922634.
- [N6] Máté Siket, Lehel Dénes-Fazakas, Levente Kovács, and György Eigner. “Numba-accelerated parameter estimation for artificial pancreas applications”. In: *2022 IEEE 20th Jubilee International Symposium on Intelligent Systems and Informatics (SISY)*. 2022, pp. 279–284. DOI: 10.1109/SISY56759.2022.10036259.
- [N7] Orsolya Csiszár, Luca Sára Pusztaházi, Lehel Dénes-Fazakas, Michael S. Gashler, Vladik Kreinovich, and Gábor Csiszár. “Uninorm-like parametric activation functions for human-understandable neural models”. In: *Knowledge-Based Systems* 260 (2023), p. 110095. ISSN: 0950-7051. DOI: <https://doi.org/10.1016/j.knosys.2022.110095>. URL: <https://www.sciencedirect.com/science/article/pii/S0950705122011911>.
- [N8] Barbara Simon, Ádám Hartveg, Lehel Dénes-Fazakas, György Eigner, and László Szilágyi. “Translating Hungarian language dialects using natural language processing models”. In: *2023 IEEE 23rd International Symposium on Computational Intelligence and Informatics (CINTI)*. 2023, pp. 000309–000316. DOI: 10.1109/CINTI59972.2023.10382068.
- [N9] Lehel Dénes-Fazakas, László Szilágyi, György Eigner, Olga Kosheleva, Martine Ceberio, and Vladik Kreinovich. “Which Activation Function Works Best for Training Artificial Pancreas: Empirical Fact and Its Theoretical Explanation”. In: *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*. 2023, pp. 496–500. DOI: 10.1109/SSCI52147.2023.10371804.

- [N10] László Szilágyi, Ágnes Györfi, Lehel Dénes-Fazakas, Szabolcs Csaholczi, Ioan-Marius Pisak-Lukáts, and Levente Kovács. “Challenges and Difficulties of Multi-Spectral MRI Based Brain Tumor Detection and Segmentation”. In: *2023 1st International Conference on Health Science and Technology (ICHST)*. 2023, pp. 1–6. DOI: 10.1109/ICHST59286.2023.10565353.
- [N11] Lehel Dénes-Fazakas, Szabolcs Csaholczi, György Eigner, Levente Kovács, and László Szilágyi. “Using Resizing Layer in U-Net to Improve Memory Efficiency”. In: *System Dependability - Theory and Applications*. Ed. by Wojciech Zamojski, Jacek Mazurkiewicz, Jarosław Sugier, Tomasz Walkowiak, and Janusz Kacprzyk. Cham: Springer Nature Switzerland, 2024, pp. 38–48. ISBN: 978-3-031-61857-4.
- [N12] Barbara Simon, Ádám Hartveg, Lehel Dénes-Fazakas, György Eigner, and László Szilágyi. “Advancing Medical Assistance: Developing an Effective Hungarian-Language Medical Chatbot with Artificial Intelligence”. In: *Information* 15.6 (2024). ISSN: 2078-2489. DOI: 10.3390/info15060297. URL: <https://www.mdpi.com/2078-2489/15/6/297>.
- [N13] Máté Siket, András Nándor Kis, Lehel Dénes-Fazakas, György Eigner, and Levente Kovács. “Database for storing and accessing diabetes related data in a standardized way”. In: *2023 IEEE 23rd International Symposium on Computational Intelligence and Informatics (CINTI)*. 2023, pp. 000285–000290. DOI: 10.1109/CINTI59972.2023.10381957.
- [N14] Barbara Simon, Ádám Hartveg, Máté Siket, Lehel Dénes-Fazakas, György Eigner, Levente Kovács, et al. “Data Collection Studies for the Better Understanding of Factors in Type 1 Diabetes Management”. In: *2024 IEEE 11th International Conference on Computational Cybernetics and Cyber-Medical Systems (ICCC)*. 2024, pp. 000375–000380. DOI: 10.1109/ICCC62278.2024.10582945.
- [N15] Lehel Dénes-Fazakas. *Mouse Dynamics Based Intrusion Detection System: Behavioural Artificial Intelligence*. Eliva Press, 2024. ISBN: 9789999318013.
- [N16] Lehel Dénes-Fazakas. *Hogyan gondolkodik a mesterséges intelligencián a cukorbetegségről*. Globeedit, 2023. ISBN: 9786206795490.
- [N17] Csaba Potyok, Barbara Simon, Ádám Hartveg, Máté Siket, Lehel Dénes-Fazakas, György Eigner, et al. “Mobile Application Development for Diabetes Patient”. In: *2024 IEEE 18th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. 2024, pp. 000559–000564.
- [N18] Miklos Nagy, Barbara Simon, László Szász, Máté Siket, Lehel Dénes-Fazakas, György Eigner, et al. “Web Application Development for Diabetes Patients”. In: *2024 IEEE 18th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. 2024, pp. 000573–000580. DOI: 10.1109/SACI60582.2024.10619716.
- [N19] Eva-H. Dulf, Alexandru George Berciu, Lehel Dénes-Fazakas, and Levente Kovacs. “Nature Inspired Optimization Algorithms in Fractional Order Controller Design”. In: *2024 IEEE 28th International Conference on Intelligent Engineering Systems (INES)*. 2024, pp. 000055–000058. DOI: 10.1109/INES63318.2024.10629099.
- [N20] Moraru Andrei, Eva-H. Dulf, Lehel Dénes-Fazakas, and Levente Kovacs. “Human Body Motion Tracking for Rehabilitation”. In: *2024 IEEE 28th International Conference on Intelligent Engineering Systems (INES)*. 2024, pp. 000209–000214. DOI: 10.1109/INES63318.2024.10629141.

- [N21] Lehel Dénes-Fazakas, László Szilágyi, Gyorgy Eigner, Olga Kosheleva, Vladik Kreinovich, and Nguyen Hoang Phuong. “Why Bump Reward Function Works Well in Training Insulin Delivery Systems”. In: *Machine Learning and Other Soft Computing Techniques: Biomedical and Related Applications*. Ed. by Nguyen Hoang Phuong, Nguyen Thi Huyen Chau, and Vladik Kreinovich. Cham: Springer Nature Switzerland, 2024, pp. 7–13. ISBN: 978-3-031-63929-6. DOI: 10.1007/978-3-031-63929-6_2. URL: https://doi.org/10.1007/978-3-031-63929-6_2.
- [N22] László Szász, Barbara Simon, Lehel Dénes-Fazakas, László Szilágyi, Levente Kovács, and György Eigner. “Advancing Personalized Diabetes Management: Enabling Research-Driven Closed-Loop Control with AndroidAPS”. In: *IFAC-PapersOnLine* 58.24 (2024). 12th IFAC Symposium on Biological and Medical Systems BMS 2024, pp. 251–256. ISSN: 2405-8963. DOI: <https://doi.org/10.1016/j.ifacol.2024.11.045>. URL: <https://www.sciencedirect.com/science/article/pii/S2405896324021724>.
- [N23] Marcell Szántó, Lehel Dénes-Fazakas, Erick Noboa, Levente Kovács, Döníz Borsos, György Eigner, et al. “Developing a Health Support System to Promote Care for the Elderly”. In: *Sensors* 25.2 (2025). ISSN: 1424-8220. DOI: 10.3390/s25020455. URL: <https://www.mdpi.com/1424-8220/25/2/455>.
- [N24] László Szilágyi, György Eigner, Levente Kovács, and Dénes-Fazakas Lehel. “Elevating Security: Mouse Dynamics in Behavior Biometrics for User Identity Authentication”. In: *2024 IEEE 6th International Symposium on Logistics and Industrial Informatics (LINDI)*. 2024, pp. 000227–000232. DOI: 10.1109/LINDI63813.2024.10820440.
- [N25] Lehel Dénes-Fazakas, Kata Sándor-Rokaly, Laszló Szász, Henrik Csuzi, Levente Kovács, László Szilágyi, et al. “Exploring the Integration of Differential Equations in Neural Networks: Theoretical Foundations, Applications, and Future Directions”. In: *2024 IEEE 24th International Symposium on Computational Intelligence and Informatics (CINTI)*. 2024, pp. 221–226. DOI: 10.1109/CINTI63048.2024.10830908.