Evaluation of Dynamic Time Warping as a Metric for Classification in Medical Data Analysis

Bachelor's Thesis

Alina Paulin Klawon
450656

20.11.2024

Supervisor: Prof. Dr. Benjamin Blankertz
Prof. Dr. Klaus-Robert Müller

Technische Universität Berlin
School of Electrical Engineering and Computer Science
Institute of Software Engineering and Theoretical Computer Science
Neurotechnology

Kurzfassung

In vielen medizinschen Anwendungsbereichen werden durch die Analyse medizinischer Zeitreihendaten wertvolle Erkenntnisse für Forschung und klinische Praxis gewonnen. Oftmals wird dabei die Ähnlichkeit zwischen Zeitreihen untersucht, um diese zu klassifizieren. Medizinische Zeitreihendaten können jedoch zeitliche Variationen aufweisen, die bei der Ähnlichkeitbewertung in vielen Anwendungsbereichen berücksichtigt werden müssen.
Die Dynamische Zeitnormierung (DTW) misst die Ähnlichkeit zwischen Sequenzen und berücksichtigt zeitliche Verschiebungen und Unterschiede in den Sequenzlängen.
In dieser Bachelorarbeit haben wir DTW als Distanzmetrik zur Klassifikation mit SVC und SVR von medizinischen Zeitreihendaten untersucht. Wir haben 8 Zeitreihen-Features aus perioperativen EEG-Aufzeichnungen verwendet, um eine Deliriumdiagnose und das Alter von Patient*innen vorherzusagen. Zusätzlich haben wir stündlich aggregierte Glukose- Herzfrequenz- und Blutdruck-Zeitreihen verwendet, um eine Diabetesdiagnose und das Alter von Patient*innen vorherzusagen. Zum Vergleich wurden statistische Baseline-Modelle trainiert.
Wir haben zwei Zeitperioden der Operationen untersucht und mit festen 4-Sekunden-Fenstern wurden in der Deliriumklassifikation höhere AUC-Werte für das Ende der Operation als für den Anfang der Anästhesie erreicht. Für das Ende der Operation erzielten alle Features mit DTW als Distanzmetrik höhere AUC-Werte als die statistischen Vergleichsmodelle. Für den Anfang der Anästhesie erreichte alpha power den höchsten AUC-Wert von 0.63 und für das Ende der Operation erreichte coherence den höchsten AUC-Wert von 0.66. Ein Feature-Vektor, bestehend aus alpha peak, aperiodic exponent und delta power, erzielte einen AUC von 0.69. Unter den getesteten Aggregierungsansätzen erzielten feste 4-Sekunden-Fenster die besten AUC-Werte. In der Altersvorhersage mit EEG-Features wurde ein minimaler MAE von 4.07 Jahren erreicht, bei eine Altersspanne von 23 Jahren.
Der maximale AUC-Wert in der Diabetesklassifikation betrug 0.67, während ein schnelleres, statistisches Vergleichsmodell einen AUC von 0.79 erreichte. In der Altersvorhersage mit Herzfrequenz- und Blutdruck-Zeitreihen wurde ein MAE von 13.25 Jahren erreicht, bei einer Altersspanne von 75 Jahren.

Abstract

The analysis of medical time series data can yield valuable insights for research and improvements in clinical practice. In many medical domains, evaluating the shape of these time series can provide significant predictive information for classification tasks. However, medical time series often exhibit timing variations, which must be addressed when assessing shape similarity in many applications.
The dynamic time warping (DTW) algorithm provides a technique for measuring similarity between sequences while accounting for timing variations and differences in sequence length.
In this thesis, we evaluated DTW as a metric for classification of medical time series data, using SVC and SVR. We utilized 8 time series features derived from perioperative EEG recordings, aiming to predict delirium diagnosis and patient age. In addition, we used hourly aggregated glucose, heart rate and blood pressure time series, aiming to predict diabetes diagnosis and patient age. For comparison, we trained statistical baseline models.
In delirium classification, we compared two distinct operative periods and observed notably higher AUC scores for the end of operation than for the start of anesthesia, when using fixed 4-second windows. For the end of operation, all features achieved higher AUC scores when using DTW as distance metric compared to the statistical baseline models. The best-performing feature, coherence, attained an AUC of 0.66 for the end of operation. In addition, alpha power achieved an AUC of 0.63 for the start of anesthesia. A combined feature, including alpha peak, aperiodic exponent and delta power, yielded an AUC of 0.69. We examined different windowing approaches and found that fixed 4-second windows yielded the highest AUC scores. In age prediction with EEG features, we observed a minimum MAE of 4.07 years, across an age range of 23 years.
In diabetes classification, we attained a maximum AUC score of 0.67, while a much faster baseline model yielded an AUC of 0.79. In age prediction with heart rate and blood pressure time series, we observed a minimum MAE of 13.25, across an age range of 75 years.

Contents

1 Introduction **1**

2 Datasets **3**
2.1 ePOD . . . 3
2.2 MIMIC-III . . . 5

3 Methodology **7**
3.1 Dynamic Programming . . . 7
3.2 Dynamic Time Warping . . . 8
3.2.1 Optimal Alignment . . . 10
3.2.2 Alignment Constraints . . . 11
3.2.3 DTW with Dynamic Programming . . . 12
3.2.4 Time and Space Complexity . . . 19
3.2.5 Sakoe-Chiba Band . . . 19
3.3 Machine Learning Models . . . 21
3.3.1 Support Vector Classification . . . 21
3.3.2 Support Vector Regression . . . 22
3.4 Metrics . . . 23
3.4.1 Area Under the ROC Curve . . . 23
3.4.2 Mean Absolute Error . . . 23
3.5 Classification & Regression Pipeline . . . 23
3.5.1 Baseline Models . . . 24

4 Results **25**
4.1 ePod Results . . . 25
4.2 MIMIC-III Results . . . 38

5 Discussion **41**
5.1 Discussion of ePod Results . . . 41
5.2 Discussion of MIMIC-III Results . . . 42

6 Conclusion **45**

List of Figures

3.1 An optimal alignment of two example sequences X and Y. 8
3.2 Two example sequences X and Y. 9
3.3 The initialized cost matrix D of the example sequences from Figure 3.2. This figure was inspired by illustrations in [1]. 13
3.4 The state of cost matrix D during the computation of the value for example cell $D[4, 5]$, highlighting the already computed, potential predecessor cells. This figure was inspired by illustrations in [1]. 14
3.5 Case $i = 1$ and $j = 1$ 15
3.6 Case $i = 1$ and $j > 1$ 15
3.7 Case $i > 1$ and $j = 1$ 15
3.8 Fully computed cost matrix D with the minimum alignment cost in cell $D[6, 7]$. This figure was inspired by illustrations in [1]. 16
3.9 The warp path in the cost matrix after backtracking. This figure was inspired by illustrations in [1]. 18
3.10 An optimal alignment of the two example sequences X and Y. 18
3.11 The Sakoe-Chiba band with a default window size of $w = 2$ in the cost matrix of the example sequences from Figure 3.2. 20

4.1 ROC curves with standard deviation over 10 cross-validation repeats of alpha power (the best-performing feature for the start of anesthesia) and its baseline model, using fixed 4-second windows and a band size of $w = 15$. 27
4.2 ROC curves with standard deviation over 10 cross-validation repeats of alpha peak and its baseline model for the start of anesthesia, using fixed 4-second windows and a band size of $w = 15$. 28
4.3 ROC curves with standard deviation over 10 cross-validation repeats of aperiodic exponent and its baseline model for the start of anesthesia, when limiting the number of windows to 20 and with a band size of $w = 5$. 29
4.4 ROC curves with standard deviation over 10 cross-validation repeats of coherence and its baseline model for the end of operation, using fixed 4-second windows and a band size of $w = 10$. 31
4.5 ROC curves with standard deviation over 10 cross-validation repeats of alpha peak and its baseline model for the end of operation, using fixed 4-second windows and a band size of $w = 10$. 32
4.6 ROC curves with standard deviation over 10 cross-validation repeats of aperiodic exponent and its baseline model for the end of operation, using fixed 4-second windows and a band size of $w = 15$. 33

4.7 MAE and standard deviation over 10 cross-validation repeats of the 3 best-performing features (using a window limit of 20 and $w = 15$) and the best baseline model for the start of anesthesia. 35
4.8 MAE and standard deviation over 10 cross-validation repeats of the 3 best-performing features (with fixed 4-second windows and $w = 5$) and the best baseline model for the end of operation. 36
4.9 ROC curves with standard deviation over 10 cross-validation repeats, comparing minimum lengths 24 and 36 as well as the baseline model, using a band size of $w = 15$. . 38
4.10 MAE scores with standard deviation over 10 cross-validation repeats in age prediction using mean heart rate recordings, comparing different minimum lengths and band sizes. 39
4.11 MAE scores with standard deviation in age prediction using mean blood pressure recordings, comparing different minimum lengths and band sizes. 39

List of Tables

2.1 EEG features. 4
2.2 Statistical variation in time series length when using fixed 4-second windows, comparing the start of anesthesia and the end of operation. 4
2.3 Statistical variation in time series length and amount of missing entries removed for each MIMIC-III feature and both minimum length constraints. 6

4.1 Mean μ and standard deviation σ of AUC scores across all 8 EEG features for the beginning of anesthesia, using different windowing approaches and Sakoe-Chiba band sizes. 25
4.2 AUC scores of alpha power for the beginning of anesthesia, comparing different windowing approaches and band sizes. The AUC was calculated over 10 cross-validation repeats. 26
4.3 AUC scores of alpha peak power and aperiodic exponent for the beginning of anesthesia, comparing different windowing approaches and Sakoe-Chiba band sizes. The AUC was calculated over 10 cross-validation repeats. 26
4.4 AUC scores of alpha peak for the beginning of anesthesia, comparing different windowing approaches and Sakoe-Chiba band sizes. The AUC was calculated over 10 cross-validation repeats. 27
4.5 Mean μ and standard deviation σ of AUC scores across all 8 EEG features for the end of operation, using different windowing approaches and Sakoe-Chiba band sizes. . . 30
4.6 AUC scores of the three best-performing features for the end of operation (coherence, alpha peak and aperiodic exponent), comparing different windowing approaches and band sizes, with fixed 4-second windows. The AUC was calculated over 10 cross-validation repeats. 30
4.7 AUC scores of a feature vector combining alpha peak, aperiodic exponent and delta power for the beginning of anesthesia, comparing different windowing approaches and Sakoe-Chiba band sizes. The AUC was calculated over 10 cross-validation repeats. 34
4.8 Mean μ and standard deviation σ of MAE scores across all 8 EEG features for the start of anesthesia, using different windowing approaches and Sakoe-Chiba band sizes. 35
4.9 Mean μ and standard deviation σ of MAE scores across all 8 EEG features for the end of operation, using different windowing approaches and Sakoe-Chiba band sizes. 36
4.10 Average MAE scores across all 8 EEG features during the end of operation, using a Sakoe-Chiba band size of $w = 10$. The table compares results based on windowing approach and DTW distance normalization approach. 37
4.11 Average time in seconds to compute a pair of Gram matrices for training and testing for the end of operation, using a band size of $w = 10$. The table compares results based on windowing approach and DTW distance normalization approach. 37

4.12 AUC scores for diabetes classification, comparing different minimum sequence lengths and Sakoe-Chiba band sizes. The AUC was calculated over 10 cross-validation repeats. 38
4.13 MAE scores with standard deviation over 10 cross-validation repeats for heart rate and blood pressure time series. The table compares the results of the DTW-based models (with minimum length 36 and $w = 5$) against the baseline models. 40

1 Introduction

In modern medicine, the analysis of medical data has become increasingly important for identifying and understanding medical correlations [2]. With the introduction of electronic health records (EHCs) and advancements in medical monitoring technology, large amounts of data have been collected over the past few decades, particularly in hospitals and biomedical research settings. The systematic evaluations of this data can help to improve the accuracy of diagnoses, clinical decision-making processes and treatment strategies [2].

Medical datasets frequently include time series data to record important medical parameters over time. In intensive care units (ICUs), continuous monitoring generates time series data, such as recordings of heart rate and blood pressure, and in anesthesia, electroencephalograms (EEGs) are used to capture the brain's electrical activity over time [3].
In many medical domains, analyzing the shape of these time series provides important predictive information for classification tasks. In cardiology, for instance, the analysis of electrocardiogram (ECG) waveforms is essential for identifying irregularities in heartbeat rhythm [4].
Medical time series can often data exhibit timing variations. Parameters such as a patient's glucose levels are collected over different durations and at varying frequencies, while heartbeats show natural variations in rhythm [5]. These timing variations must be addressed when assessing shape similarity in many medical domains, requiring a suitable analysis technique [6], [7].

The dynamic time warping (DTW) algorithm presents a technique to determine shape similarity between sequences while accounting for timing variations and differences in sequence length [8]. DTW uses a non-linear function to effectively stretch and compress the time axes of two sequences, aligning the sequences and capturing similarities in shape despite shifts in timing [8]. It determines the minimum cost of aligning the sequences and the normalized minimum alignment cost can be used as a measure of similarity, where a small distance indicates that two sequences are similar in shape [9].

Over the past few decades, numerous studies have demonstrated that DTW is an effective metric for classification of time series data in medical data analysis [8], [6]. Originally developed as a distance metric for spoken word recognition, the algorithm has since been adapted for various applications across different domains, including gesture recognition, bioinformatics and medicine [9], [10], [11], [8].
In 2002, B. Huang and W. Kinsner applied DTW in electrocardiogram frame classification, aiming to improve ECG signal "compression, browsability [for faster analysis] and accurate recognition of cardiac rhythms for defibrillators" [6]. To account for natural, nonlinear timing variations in heartbeat rhythms, the authors used DTW to align the signals and address these fluctuations. Instead of directly using the total accumulated distance between a frame and a template, they assessed similarity with the mean of absolute residuals along the alignment path. Testing on a 10-minute ECG recording achieved a residual error of 1.33%, identifying DTW as an effective metric for classification in this context [6].

Another 2018 study by Y. Shi et al. applied DTW to improve the analysis of EEG signals for recognizing motor imagery tasks, which are mental activities where a person imagines movements, such as moving an arm [7]. Accurately identifying these imagined movements from EEG signals is important for brain-computer interfaces, which allow individuals with disabilities to control devices through thought [12]. The authors combined Wavelet Packet Decomposition for improved feature extraction with DTW to measure signal similarity. Quadratic discriminant analysis was used as classification method and testing showed that classification accuracy improved from 83.53% (using traditional methods) to 90.89% [7].

In this thesis, we aim to further evaluate the applicability and effectiveness of DTW as a distance metric for classification of medical time series data. More specifically, we assess the use of DTW in support vector classification (SVC) and support vector regression (SVR), using features derived from the ePOD study and the MIMIC-III database.

To conduct this evaluation, we begin by introducing the ePOD study and MIMIC-III database in chapter 2, along with the MIMIC-Extract pipeline and the selected features and target variables.
Chapter 3 outlines the methodology, starting with the underlying concept of dynamic programming and then presenting the DTW algorithm. This section provides a more formal description of the DTW problem, defining an optimal alignment and covering the alignment constraints. We then describe the implementation of DTW with dynamic programming and outline the importance of normalizing the DTW distance. We also cover the computational complexity of DTW and introduce the Sakoe-Chiba band constraint as a technique to improve the algorithm's efficiency. We present the machine learning algorithms employed, specifically SVC and SVR, along with the metrics used to evaluate predictive performance. Finally, we describe our data preprocessing approach and specify the parameters used in SVC and SVR. Additionally, we present the baseline models trained for comparison with the DTW-based models.
Chapter 4 presents the results attained for the ePod and MIMIC-III features.
These results are discussed in chapter 5, where we interpret the observed scores, highlight interesting results, compare our results with existing literature, address the limitations of this thesis and outline its contributions.
Chapter 6 summarizes the results, conclusions and implications of this thesis and outlines open questions for future research.

2 Datasets

2.1 ePOD

The ePOD study included 287 elderly patients treated at the Department of Anesthesiology and Intensive Care Medicine at Charité Universitätsmedizin Berlin. The objective of the study was to analyze how characteristics of perioperative electroencephalography recordings might be connected to the occurrence of delirium in elderly patients after surgery [13].
The study's inclusion criteria required participants to be over the age of 70 years, to be scheduled for operations lasting more than 1 hour and to be able to give informed consent. Anesthesia was induced and maintained using either Propofol or an inhalative anesthetic such as Sevoflurane or Desflurane. EEG recordings were obtained with a frontal EEG monitor (SEDline Root Monitor©, Masimo Corporation, USA) "from the beginning of the induction of anesthesia to the intraoperative setting up until 1 [hour] in the recovery room" [13]. Postoperative delirium (POD) was assessed using the Nursing Delirium Screening Scale (NU-DESC) for up to 5 days after the surgery. Prior to its start, ePOD was registered in the clinicaltrials.gov database (NCT03879850) [13].

We examined two distinct time periods of the operations:

1. The beginning of the anesthesia, starting at the induction, including the intubation and the loss of consciousness.

2. The end of the operation, starting at the end of anesthesia when medication was stopped until wake up.

These time periods varied in length for each patient.

For each period, we analyzed time series of extracted features from the EEG data and the EEG spectrum. The spectrum was calculated with the psd_array_welch() function of the MNE library [14]. To generate the time series, each period was divided into a series of time windows, and features were calculated for each window. These time series were generated and provided to us by the supervising group. Two windowing approaches were applied:

1. Each window had a fixed length of 4 seconds.

2. The number of windows was limited to either 20 or 50, with a minimum window length of 4 seconds. If a period was longer than 4 seconds multiplied by the maximum number of windows, the window size was increased to the length of the period divided by the maximum number of windows.

Table 2.1 presents the extracted time series features:

Feature	**Description**
alpha peak	Frequency at which alpha power reaches its maximum.
alpha peak power	Maximum power at the alpha peak frequency.
alpha power	Mean power in the frequency range $(8, 13)$.
aperiodic exponent	Exponent of the aperiodic component of the EEG signal.
aperiodic offset	Offset of the aperiodic component of the EEG signal.
beta power	Mean power in the frequency range $(13, 30)$.
coherence	Mean coherence with the coherence between all channel pairs.
delta power	Mean power in the frequency range $(0.3, 4)$.

Table 2.1: EEG features.

The features alpha peak, alpha peak power, aperiodic exponent and aperiodic offset were calculated with the fooof library, while coherence was derived with the coherence() function of scipy.signal [15], [16].

Patients with missing markers for the start of anesthesia or the end of operation were excluded from the analysis. For the start of anesthesia, 55 out of 179 included patients were diagnosed with delirium. For the end of operation, 52 out of 174 included patients were diagnosed with delirium. The ages of the included patients ranged from 70 to 92, with a mean age of 77 years and a standard deviation of approximately 5 years.
Notably, when using fixed 4-second windows, time series included for the end of the operation had a much higher mean length of 186.20 with a higher standard deviation of 100.28, compared to time series included for the start of anesthesia, which had a mean length of 73.16 with a standard deviation of 31.55. Table 2.2 presents the statistical variation in time series length when using fixed 4-second windows, comparing both time periods:

Windowing Approach	**Time Period**	**Time Series Length**			
		mean	**std**	**mean**	**std**
4-second windows	start of anesthesia	73.16	31.55	14	279
	end of operation	186.20	100.28	6	491

Table 2.2: Statistical variation in time series length when using fixed 4-second windows, comparing the start of anesthesia and the end of operation.

2.2 MIMIC-III

The publicly available MIMIC-III database contains de-identified electronic health records from ICU patients treated at the Beth Israel Deaconess Medical Center in Boston, Massachusetts. It "contains data associated with 53423 distinct hospital admissions for adult patients [...] admitted to critical care units between 2001 and 2012" [3].

Rather than working directly with the raw health records from MIMIC-III, we utilized the processed data structures generated by the open-source pipeline MIMIC-Extract. The pipeline extracts and pre-processes medically relevant information from the MIMIC-III database [17]. It uses SQL queries to produce multiple output tables. We utilized variables included in the patients-table, the vitals_labs-table and the codes-table.
The patients-table contains demographic variables such as the patients' age, which is masked as 300 for patients older than 89 years in the MIMIC-III database.
The vitals_labs-table includes "104 clinically aggregated time-series variables [...] related to vital signs (e.g., heart rate or blood pressure) and laboratory test results (e.g., [glucose levels])" [17]. MIMIC-Extract standardizes vital signs and laboratory test results by first converting measurements into consistent units. It handles extreme values falling outside of clinically reasonable ranges by treating them as missing. Values falling outside of more refined physiologically valid ranges are replaced with the nearest valid value. Finally, MIMIC-Extract discretizes "all extracted time series [...] into hourly [intervals]" and a predefined threshold is applied to retain only those vital signs and laboratory test results with fewer missing values [17].
In addition, the codes-table contains time series of ICD-9 codes, documenting the progression of diagnoses throughout a patient's ICU stay.
The pipeline's inclusion criteria specify that only the first ICU stay for each patient is extracted, with a minimum patient age of 15 years and a stay duration between 12 hours and 10 days [17].

We extracted mean glucose, mean heart rate and mean blood pressure recordings from the vitals_labs-table, creating the following 6 data sets:

First, we generated 2 data sets from mean glucose recordings, combined with diabetes diagnoses from the codes-table, and applied a different minimum length constraint to each data set.
MIMIC-III includes 20 ICD-9 codes that indicate various forms of diabetes (e.g. $250.0x$ for "diabetes mellitus without mention of complications" [18]). For each patient, we determined whether they had been diagnosed with *any* of these 20 types of diabetes.
After removing missing entries from the glucose time series, we applied the minimum length constraints, only retaining time series with a minimum length of either 24 or 36. All time series were sliced to a maximum length of 48.
We used the RandomUnderSampler class of the sklearn-library to generate 2 balanced data sets, each containing 2850 patients, with random_state set to 42 [19].

We then generated 2 data sets from mean blood pressure recordings and 2 data sets from mean heart rate recordings, combined with patient age, only including patients aged 89 years or younger.
After removing missing entries from the time series, we again applied the different minimum length constraints for each feature, only retaining time series with a minimum length of either 24 or 36. All

time series were sliced to a maximum length of 48. For each data set, we selected 5000 patients using the random.choice() function from the numpy library, with random.seed(42) for reproducibility [20]. The mean age of the selected patients for both the heart rate and blood pressure time series and both minimum length constraints is 63 with a standard deviation of 16 to 17 years.

MIMIC-Extract retrieved already present missing entries in MIMIC-III and generated additional missing entries when handling outliers, which we removed before creating the datasets. For each selected time series, we counted the number of missing entries we removed between the first valid entry and the last valid entry (up to a maximum of 48 valid entries).
For instance, an average of about 70 missing entries were removed from glucose time series with minimum length 24. Considering the mean length of these time series of about 36 (after removing missing entries and slicing to a maximum length of 48) together with the hourly aggregation through MIMIC-Extract, these time series actually represent the mean glucose level development over an average of roughly 36 + 70 = 106 hours.
Table 2.3 shows the average time series length and the average amount of missing entries for each feature and minimum length constraint:

Feature	**Min. Length**	**Time Series Length**		**Amount of NaN-Entries**	
		mean	**std**	**mean**	**std**
glucose	24	35.98	8.70	69.86	49.69
	36	44.37	4.36	71.51	52.17
heart rate	24	41.83	8.49	2.21	4.26
	36	46.31	3.29	2.42	4.42
blood pressure	24	41.84	8.43	3.47	5.82
	36	46.34	3.29	3.47	5.30

Table 2.3: Statistical variation in time series length and amount of missing entries removed for each MIMIC-III feature and both minimum length constraints.

3 Methodology

3.1 Dynamic Programming

In 1953, Richard Bellman introduced dynamic programming as an approach for solving optimization problems that can be decomposed into *similar* subproblems [21]. The underlying idea of dynamic programming is to avoid redundant computations by storing the solutions to subproblems that have already been solved [21].

Definition 3.1.1 As explained in [22], the objective in an optimization problem is to find an optimal solution from a set of feasible solutions. An optimization problem is commonly defined by an objective function f and a set of constraints C. The objective function $f : S \to \mathbb{R}$ maps each feasible solution $x \in S$ to a real number, with $S \subseteq \mathbb{R}^n$ and $n \in \mathbb{N}$. S represents all solutions that satisfy the constraints in C. The goal is to find a solution $x^* \in S$ that satisfies the constraints and minimizes (or maximizes) the objective function f [22].

To solve an optimization problem with dynamic programming, a recursive formula is first defined, as described in [21]. This formula specifies how the solution to the original problem is constructed from the solutions to its subproblems.
There are two common approaches to dynamic programming: the top-down approach and the bottom-up approach [21].
DTW uses the bottom-up approach, in which the original problem is solved *iteratively* by solving the most simple subproblems first [23], [21]. In the bottom-up approach, the recursive formula is repeatedly applied to combine the already computed solutions of subproblems, generating solutions to the next bigger subproblems. The solutions are stored in a tabulation data structure, allowing subsequent subproblems to efficiently access these precomputed values. In contrast to the top-down approach, *all* possible subproblems are systematically evaluated in the bottom-up approach [21].

In many applications that use dynamic programming, the recursive formula is only used to calculate the optimal value to an optimization problem. To identify the specific decisions that correspond with this value, it is oftentimes necessary to retrace the decisions with backtracking [21].

3.2 Dynamic Time Warping

Dynamic Time Warping, introduced by Hiroaki Sakoe and Seibi Chiba in 1978, provides a technique to measure the similarity of sequences, such as time series, in regard to their shapes [9], [8]. The algorithm uses dynamic programming to determine the similarity of sequence shapes while accounting for timing variations and differences in sequence length. [9].

Originally, dynamic time warping was designed as a distance metric for sequences in spoken word recognition, with the aim to identify spoken words by comparing the shape of recordings [9]. However, as people speak at varying speeds and with different rhythms, timing inconsistencies commonly arise within these sequences, making these variations a central challenge in spoken word recognition [9].

DTW addresses this challenge and essentially stretches and compresses the time axes of the sequences to capture similarities in shape despite shifts in timing, as described in [8]. It achieves this through a non-linear function (or warp path) that maps the indices of sequence X to the indices of sequence Y, allowing each index of both sequences to be matched with multiple indices of the other sequence [8]. The algorithm determines the cost of an optimal alignment, which minimizes the total alignment cost. Finally, the normalized minimum cost (or distance) can be used as a measure of similarity, where a small distance indicates that two sequences are similar. [9].

Figure 3.1 shows an optimal alignment of two sequences X and Y, computed with DTW. To capture shape similarities despite shifts in timing, some elements are aligned with multiple elements from the other sequence:

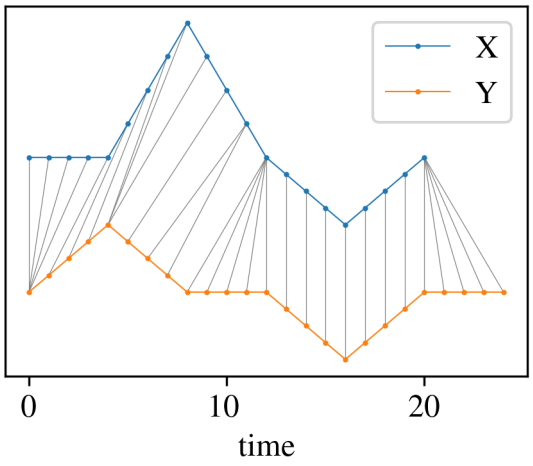

Figure 3.1: An optimal alignment of two example sequences X and Y.

Definition 3.2.1 In the following sections, we use the notation $[K]$, with $K \in \mathbb{N}$, to denote the set of all natural numbers from 1 to K, inclusive:

$$[K] = \{1, 2, 3, \ldots, K-1, K\} \tag{3.1}$$

The notation $i \in [K]$ means that i is an integer such that $1 \leq i \leq K$. It is commonly used to denote an index set ranging from 1 to K.

Definition 3.2.2 In the context of this thesis, we define a sequence X as a function $X : [a, b] \to \mathbb{R}^d$, which maps an interval of the natural numbers $[a, b] \subseteq \mathbb{N}$ to the d-dimensional space $\mathbb{R}^d$, with $d \in \mathbb{N}$ and $a \leq b$. A sequence X can be notated as:

$$X = x_1, x_2, \ldots, x_N \quad N \in \mathbb{N} \tag{3.2}$$

where $x_i \in \mathbb{R}^d$ for each $i \in [a, b]$. $N = b - a + 1$ is the number of elements in the sequence.

Input

The DTW algorithm takes two temporal sequences X and Y as input, which may differ in length [23]:

$$\begin{aligned} X &= x_1, x_2, \ldots, x_N \qquad N \in \mathbb{N} \\ Y &= y_1, y_2, \ldots, y_M \qquad M \in \mathbb{N} \end{aligned} \tag{3.3}$$

X and Y are sequences of either scalars or vectors $x_i, y_j \in \mathbb{R}^n$, with $i \in [N]$, $j \in [M]$ and $n \in \mathbb{N}$.

Figure 3.2 provides an example of two temporal sequences X and Y, where X is of length $N = 6$ and Y is of length $M = 7$:

$$\begin{aligned} X &= 0, 0, 2, 0, -1, 0 \\ Y &= 0, 1, 0, 0, -1, 0, 0 \end{aligned} \tag{3.4}$$

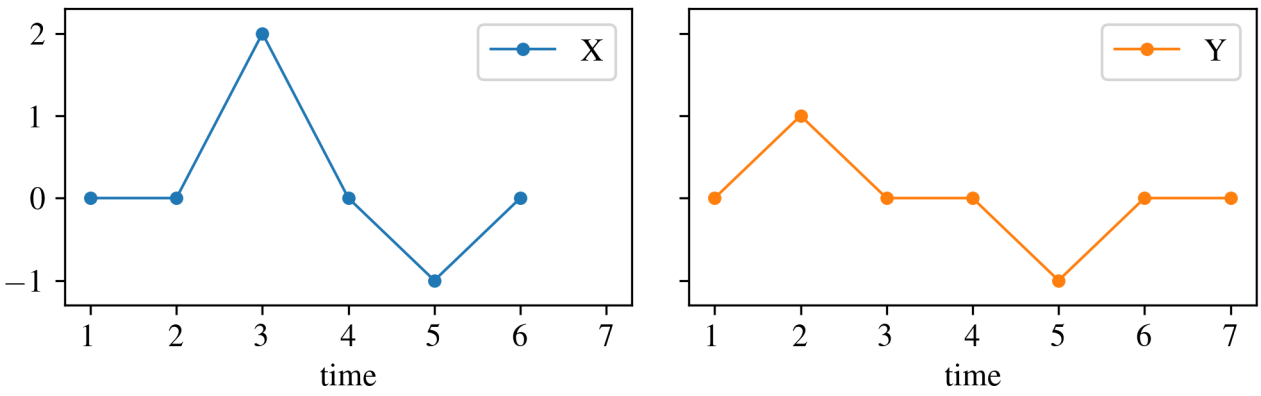

Figure 3.2: Two example sequences X and Y.

3.2.1 Optimal Alignment

To align two sequences as cost-effectively as possible, the algorithm computes an optimal warp path [23].

Definition 3.2.3 A warp path W is a sequence of tuples:

$$\begin{aligned} W &= w_1, w_2, \ldots, w_K & \max(N, M) \leq K < N + M \wedge K \in \mathbb{N} \\ w_k &= (i, j) & k \in [K] \wedge i \in [N] \wedge j \in [M] \end{aligned} \tag{3.5}$$

A tuple $w_k = (i, j)$ indicates that the i-th element of sequence X is aligned with the j-th element of sequence Y. W is of length K with $\max(N, M) \leq K$ and $K < N + M$, as described in [23].

Definition 3.2.4 A warp path W is considered *optimal* if it minimizes the total alignment cost. The cost of a warp path (or the total alignment cost) is calculated by summing the distances between the aligned elements of X and Y [23]:

$$\text{dist}(W) = \sum_{k=1}^{K} \text{d}\big(X[w_{ki}], Y[w_{kj}]\big) \tag{3.6}$$

In this equation, $\text{d}\big(X[w_{ki}], Y[w_{kj}]\big)$ is the distance between the i-th element of sequence X and the j-th element of sequence Y, as specified by the k-th tuple in the warp path W [23].

A commonly used metric for calculating the distance between elements of X and Y is the Euclidean distance:

Definition 3.2.5 The Euclidean distance of two points $p = (p_1, p_2, \ldots, p_n)$ and $q = (q_1, q_2, \ldots, q_n)$, with $n \in \mathbb{N}$, can be calculated as follows [24]:

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} \tag{3.7}$$

3.2.2 Alignment Constraints

To preserve essential properties of temporal sequences, an optimal warp path must satisfy the following constraints of monotonicity, continuity, and boundary values, as presented in [9]:

1. **Monotonicity**:
   The indices i of sequence X and j of sequence Y must be monotonically increasing in the warp path - reflecting the idea that time cannot reverse direction. For tuples $w_a = (i(a), j(a))$ and $w_b = (i(b), j(b))$ with $1 \leq a < b \leq K$, the following must hold:

   $$i(a) \leq i(b) \text{ and } j(a) \leq j(b) \tag{3.8}$$

2. **Continuity**:
   To preserve the continuity of the sequence's indices in the warp path, consecutive tuples $w_k = (i(k), j(k))$ and $w_{k+1} = (i(k+1), j(k+1))$ with $k \in [K-1]$ must satisfy:

   $$i(k+1) - i(k) \leq 1 \text{ and } j(k+1) - j(k) \leq 1 \tag{3.9}$$

   The condition ensures that the warp path advances in both sequences without skipping any indices.

3. **Boundary Values**:
   The warp path must start with $w_1 = (1, 1)$ and end with $w_K = (N, M)$, ensuring that the first elements and the respective last elements of both sequences are aligned.

"As a result of [the conditions of monotonicity and continuity], the following relation holds between two consecutive points", defining only three possible predecessors of a tuple w_k [9]:

$$w_{k-1} = \begin{cases} & (i(k)-1, j(k)-1), \\ \text{or} & (i(k)-1, j(k)), \\ \text{or} & (i(k), j(k)-1) \end{cases} \tag{3.10}$$

3.2.3 DTW with Dynamic Programming

DTW uses a dynamic programming approach to find the optimal alignment between two sequences X and Y [9]. Instead of comparing the entire sequences directly, it breaks the initial sequences down into smaller subsequences $X[1:i]$ and $Y[1:j]$, with $i \in [N]$ and $j \in [M]$. The algorithm computes and stores the minimum alignment costs for these smaller subsequences [23]. It then uses the relation of consecutive tuples (Equation 3.10) as a recursive formula to derive the minimum alignment cost for the initial sequences from the intermediate solutions. Finally, backtracking is used to compute the corresponding alignment (or optimal warp path) of X and Y.

Cost Matrix

The data structure used in the dynamic programming approach is a cost matrix $D \in \mathbb{R}^{(N+1)\times(M+1)}$, where each cell $D[i,j]$ of the matrix is used to store the minimum cost of aligning the subsequences $X[1:i]$ and $Y[1:j]$ [23].

Algorithm 3.1 summarizes the initialization and computation of the cost matrix D, as shown in [25]:

Algorithm 3.1 Cost_Matrix(X, Y)

1. Initialize $N \leftarrow \text{length}(X)$ and $M \leftarrow \text{length}(Y)$
2. Initialize the cost matrix $D \in \mathbb{R}^{(N+1)\times(M+1)}$ with infinity and set $D[0][0] \leftarrow 0$
3. **For** $i \leftarrow 1$ **to** N **do**
   a) **For** $j \leftarrow 1$ **to** M **do**
      i. $D[i][j] \leftarrow \text{d}(X[i], Y[j]) + \min \begin{cases} D[i-1][j-1], \\ D[i-1][j], \\ D[i][j-1] \end{cases}$
4. **Return** D

Cost Matrix Initialization

The cost matrix D is first initialized by filling the entire matrix with infinity. $D[0,0]$ is then set to 0 to ensure an initial alignment cost of 0. The algorithm utilizes an additional row 0 and column 0 to handle boundary cases more conveniently, resulting in the size of the matrix being $(N+1) \times (M+1)$.

Figure 3.3 shows the initialized cost matrix D for the example sequences X and Y from Figure 3.2. Given that sequence X has a length of 6 and sequence Y has a length of 7, the resulting cost matrix is of size $(6+1)\times(7+1) = 7\times 8$. Figure 3.3 highlights the additional row 0 and column 0 and illustrates how the indices of the matrix align with the indices of X and Y.

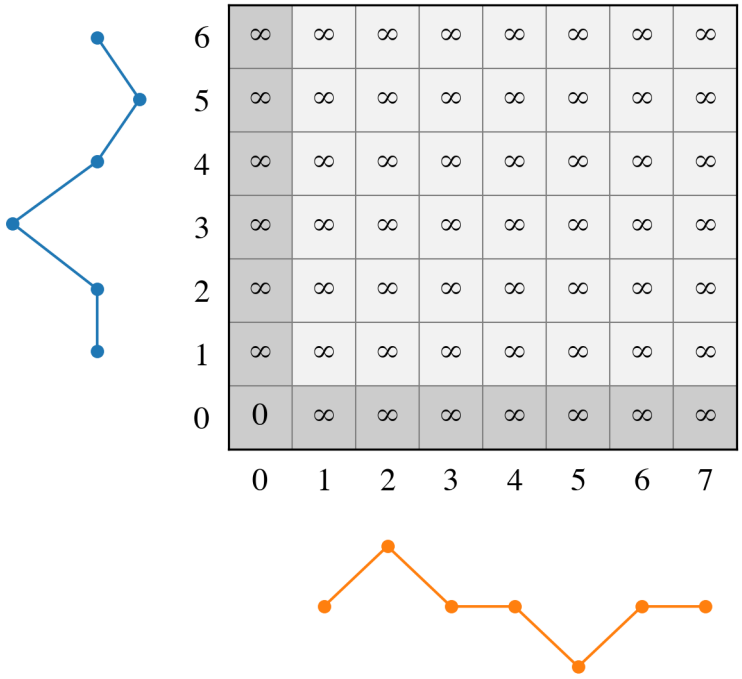

Figure 3.3: The initialized cost matrix D of the example sequences from Figure 3.2. This figure was inspired by illustrations in [1].

Cost Matrix Computation

After initializing the cost matrix, the algorithm uses a nested loop to compute the minimum alignment cost for each cell $D[i, j]$ (or each pair of subsequences $X[1 : i]$ and $Y[1 : j]$), where the row indices i correspond to the indices of sequence X and the column indices j correspond to the indices of sequence Y [23].

The nested loop starts with $i = 1$ and $j = 1$, to first compute the minimum alignment cost of the smallest possible subsequences $X[1 : 1] = x_1$ and $Y[1 : 1] = y_1$.
The alignment cost of $X[1 : i]$ and $Y[1 : i]$ is calculated using an iterative approach based on Equation 3.10. It is determined by adding the distance between the elements $X[i]$ and $Y[i]$ to the minimum alignment cost of the three possible pairs of preceding subsequences: $X[1 : i-1]$ and $Y[1 : j-1]$ or $X[1 : i-1]$ and $Y[1 : j]$ or $X[1 : i]$ and $Y[1 : j-1]$. The minimum alignment costs of these pairs have already been computed in previous iterations of the nested loop and are stored in $D[i-1, j-1]$, $D[i-1, j]$, and $D[i, j-1]$, respectively.

Therefore, the value of $D[i, j]$ is calculated as follows [23]:

$$D[i,j] = \mathrm{d}\big(X[i], Y[j]\big) + \min \begin{cases} D[i-1, j-1], \\ D[i-1, j], \\ D[i, j-1] \end{cases} \tag{3.11}$$

Cost Matrix Computation Example

Figure 3.4 illustrates the state of the cost matrix D during the computation of the value for example cell $D[4, 5]$. This value represents the minimum cost of aligning $X[1 : 4]$ and $Y[1 : 5]$. Figure 3.4 also highlights the already computed, potential predecessor cells:

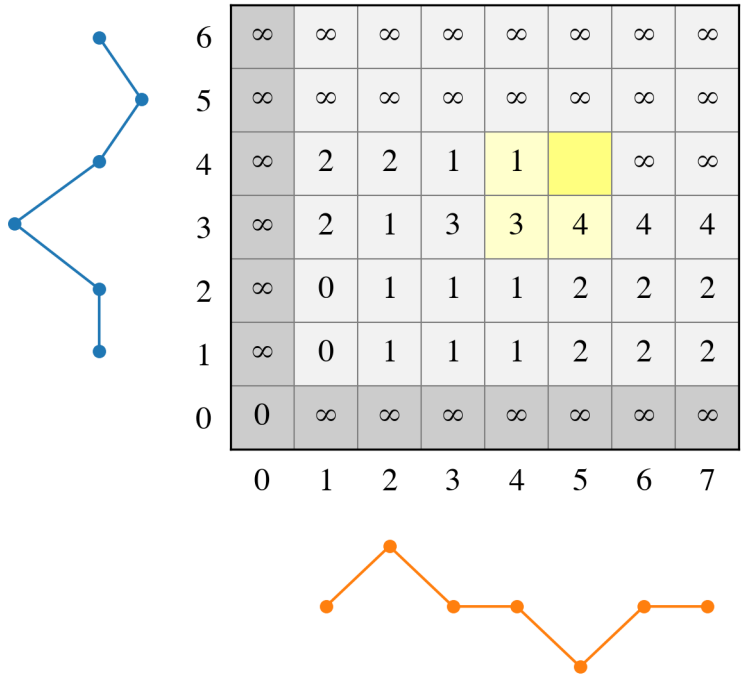

Figure 3.4: The state of cost matrix D during the computation of the value for example cell $D[4, 5]$, highlighting the already computed, potential predecessor cells. This figure was inspired by illustrations in [1].

In this example, the value of $D[4, 5]$ is determined by first computing the Euclidean distance of the 4-th element of sequence X and the 5-th element of sequence Y. The algorithm then adds the smallest value of the predecessor cells, resulting in $D[4, 5] = 2$:

$$D[4,5] = \mathrm{d}\big(X[4], Y[5]\big) + \min \begin{cases} D[3,4], \\ D[3,5], \\ D[4,4] \end{cases} = \sqrt{(0-(-1))^2} + \min \begin{cases} 3, \\ 4, \\ 1 \end{cases} = 1 + 1 = 2 \tag{3.12}$$

Boundary Cases

A boundary case arises when $i = 1$ or $j = 1$. The algorithm uses the additional row 0 and column 0 to handle these boundary cases. As the nested loop only iterates from $i = 1$ to N and $j = 1$ to M, the values in row 0 and column 0 remain set to infinity throughout the algorithm (except for $D[0,0] = 0$):

1. When $i = 1$ and $j = 1$, the algorithm aligns the first elements of X and Y. With $D[1,0] = \infty$, $D[0,1] = \infty$, $D[0,0] = 0$ and $0 < \infty$, the algorithm must select $D[0,0] = 0$ as predecessor cell of $D[1,1]$. This guarantees that the minimum alignment cost starts from 0 and the cost of aligning $X[1:1]$ and $Y[1:1]$ is calculated as:

$$D[1,1] = \mathrm{d}\big(X[1], Y[1]\big) + \min \begin{cases} 0, \\ \infty, \\ \infty \end{cases} = \mathrm{d}\big(X[1], Y[1]\big) + 0 = \mathrm{d}\big(X[1], Y[1]\big)$$

2. When $i = 1$ and $j > 1$, the algorithm aligns the first element of X with the first $j > 1$ elements of Y. Since there are no preceding elements in X, the only valid transition is from $D[1, j-1]$ to $D[1, j]$. Utilizing the additional row 0 ensures that the values $D[0, j-1] = \infty$ and $D[0, j] = \infty$ are always greater than $D[1, j-1]$, thereby enforcing that the algorithm selects the only valid transition from $D[1, j-1]$.

3. Similarly, when $i > 1$ and $j = 1$, the algorithm aligns the first element of Y with the first $i > 1$ elements of X and the only valid transition is from $D[i-1, 1]$ to $D[i, 1]$. The additional column 0 ensures that the values $D[i-1, 0] = \infty$ and $D[i, 0] = \infty$ are always greater than $D[i-1, 1]$, thereby ensuring that the algorithm always chooses the only valid transition from $D[i-1, 1]$.

Figures 3.5, 3.6 and 3.7 illustrate the different boundary cases:

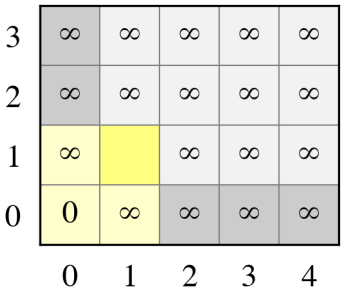

Figure 3.5: Case $i = 1$ and $j = 1$

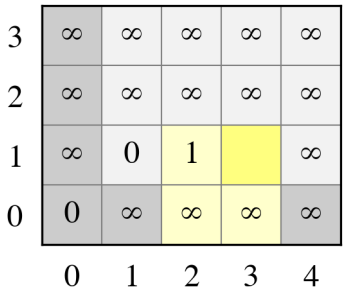

Figure 3.6: Case $i = 1$ and $j > 1$

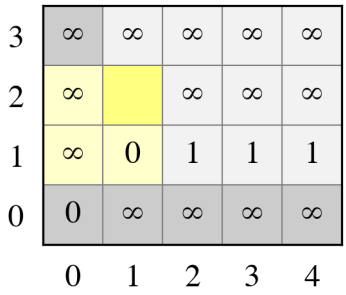

Figure 3.7: Case $i > 1$ and $j = 1$

Minimum Alignment Cost

The nested loop finally terminates when $i = N$ and $j = M$. Since $X[1:N] = X$ and $Y[1:M] = Y$, the value of $D[N, M]$ corresponds to the minimum alignment cost of the initial sequences X and Y [23].

Figure 3.8 shows the fully computed cost matrix of the example sequences X and Y, with a minimum alignment cost of $D[N, M] = D[6, 7] = 1$:

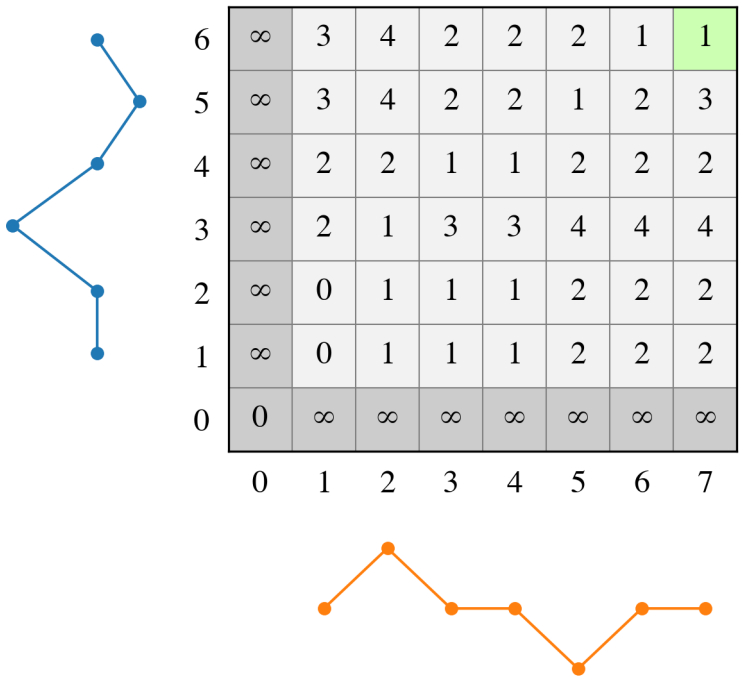

Figure 3.8: Fully computed cost matrix D with the minimum alignment cost in cell $D[6, 7]$. This figure was inspired by illustrations in [1].

Normalization of the Minimum Alignment Cost

Longer sequences contain more elements that need to be aligned. The cost of aligning these elements accumulates along the warp path. Consequently, pairs of longer sequences might naturally have higher minimum alignment costs, even when the sequences are relatively similar, as described in [26]. Since the minimum alignment cost of two sequences is influenced by sequence length, it is less comparable across different pairs of sequences with varying lengths. To make the costs more comparable, different normalization techniques can be applied.
One common approach to normalizing the minimum alignment cost is to divide $D[N, M]$ by the warp path length K [27]. Another, slightly faster method that does not require computing the warp path and its length, is normalization by the sum of the sequence lengths $N + M$ (an upper bound for K) [1]:

$$C = \frac{D[N, M]}{N + M} \tag{3.13}$$

Backtracking

To derive the optimal warp path W (or optimal alignment) that corresponds to the minimum alignment cost, the algorithm uses the fully computed cost matrix D and traces W backward from $D[N, M]$ to $D[1, 1]$ [23]. Algorithm 3.2 summarizes the steps to compute W, as described in [28]:

Algorithm 3.2 DTW_Alignment($D, N = \text{length}(X), M = \text{length}(Y)$)

1. Initialize $i \leftarrow N$ and $j \leftarrow M$
2. Initialize $W \leftarrow []$
3. **While** $i > 0$ **or** $j > 0$ **do**
   a) Append (i, j) to W
   b) $i, j \leftarrow \text{argmin} \begin{cases} D[i-1][j-1], & \textit{\# move diagonally} \\ D[i-1][j], & \textit{\# move down} \\ D[i][j-1] & \textit{\# move left} \end{cases}$
4. Reverse W
5. **Return** W

The backtracking algorithm takes the fully computed cost matrix D and the lengths N and M of the sequences X and Y as input.
It starts by initializing the indices $i = N$ and $j = M$, which correspond to the current positions in the sequences as the algorithm traces back through the matrix. An empty list W is initialized to store the indices of the optimal alignment between X and Y.
The algorithm then enters a loop to trace back through the matrix. In each iteration, the current index pair (i, j) is appended to W. The algorithm then updates i and j by choosing the direction that corresponds to the minimum cost among the possible moves: diagonally (down and to the left), down or to the left.
The loop terminates once the algorithm reaches the start of the sequences, at which point W is reversed to reflect the correct order of alignment. Finally, the algorithm returns W, which now contains the indices of the optimal alignment path from $D[1, 1]$ to $D[N, M]$.

Figure 3.9 shows the fully computed cost matrix of the example sequences X and Y and the warp path calculated by the backtracking algorithm. The optimal warp path of the example sequences is of length 8. Figure 3.10 finally illustrates the optimal alignment of the example sequences X and Y.

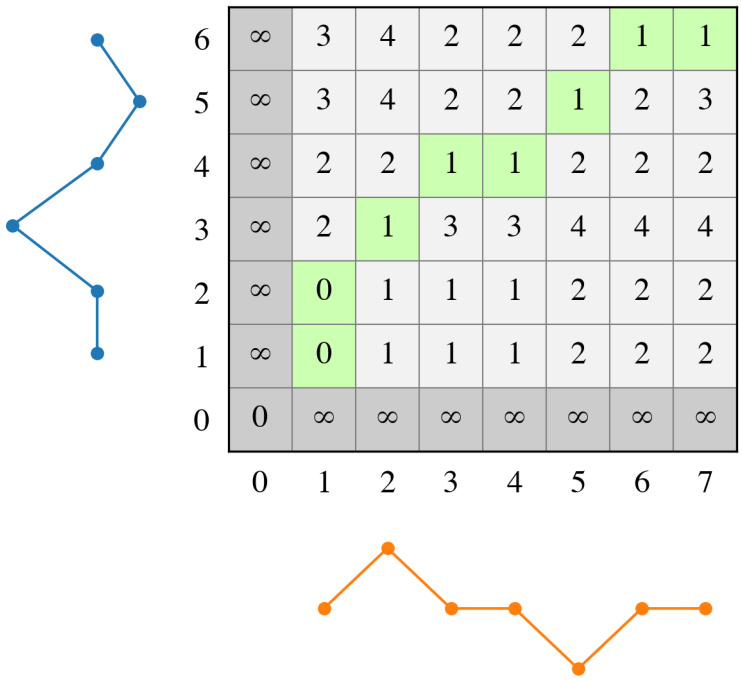

Figure 3.9: The warp path in the cost matrix after backtracking. This figure was inspired by illustrations in [1].

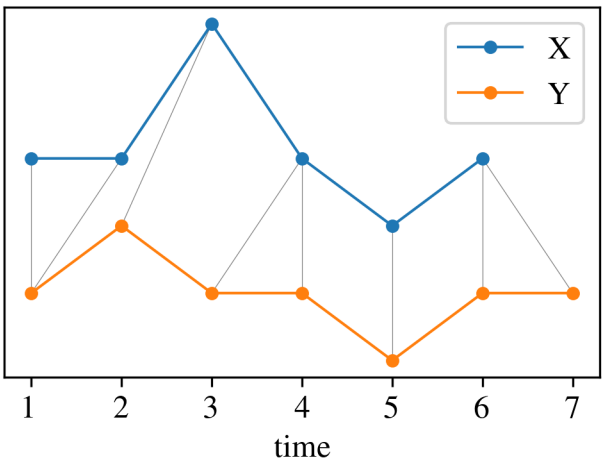

Figure 3.10: An optimal alignment of the two example sequences X and Y.

3.2.4 Time and Space Complexity

Definition 3.2.6 The *time complexity* of an algorithm is a function $T(n)$ that describes the amount of computational steps required to execute the algorithm as a function of the input size n. The Big O notation $O(T(n))$ characterizes the asymptotic upper bound of $T(n)$. This means that there exist two positive constants c and n_0 such that $T(n) \leq c \cdot f(n)$ for all $n \geq n_0$, where $f(n)$ is a non-negative function. [29] For instance, time complexities used in this section are $O(1)$, $O(n)$ and $O(n^2)$.

As described in [23], the runtime complexity of the standard DTW algorithm is $O(N \times M)$, where N is the length of input sequence X and M is the length of input sequence Y. It is largely determined by the computation of the cost matrix D. The space complexity of the DTW algorithm is also $O(N \times M)$, as the algorithm requires storing the entire cost matrix D of size $(N+1) \times (M+1)$ [23].

3.2.5 Sakoe-Chiba Band

The Sakoe-Chiba band constraint was designed to enhance the computational efficiency of DTW, as shown in [9]. The constraint is defined by a window size w that restricts how far the warp path W can deviate from the diagonal of the cost matrix, reducing the number of cost matrix elements to compute. Since the time complexity of the standard DTW algorithm is primarily determined by the computation of the cost matrix, introducing the Sakoe-Chiba band results in a more efficient computation [9]. Algorithm 3.3 summarizes the steps to calculate the cost matrix D when using the Sakoe-Chiba band with window size w:

Algorithm 3.3 Cost_Matrix_Window(X, Y, w)

1. Initialize $N \leftarrow \text{length}(X)$ and $M \leftarrow \text{length}(Y)$
2. Initialize the cost matrix $D \in \mathbb{R}^{(N+1)\times(M+1)}$ with infinity and $D[0][0] \leftarrow 0$
3. Initialize the window size $w \leftarrow \max(w, \text{abs}(N - M))$
4. **For** $i \leftarrow 1$ **to** N **do**
   a) $l \leftarrow \max(1, i - w)$ *# lower window bound*
   b) $u \leftarrow \min(M, i + w)$ *# upper window bound*
   c) **For** $j \leftarrow l$ **to** u **do**
      i. $$D[i][j] \leftarrow \text{dist}(X[i], Y[j]) + \min \begin{cases} D[i-1][j-1], \\ D[i-1][j], \\ D[i][j-1] \end{cases}$$
5. **Return** D

As input, Algorithm 3.3 takes the sequences X and Y, along with a predefined window size w. Similar to the standard DTW algorithm, it initializes N as the length of sequence X and M as the length of the sequence Y. The cost matrix D is initialized with infinity and $D[0,0]$ is set to 0. The window size w is updated as the maximum of the predefined default window size and the absolute difference between N and M to ensure the warp path can extend far enough to reach the endpoints of both sequences, even if their lengths differ significantly.
The algorithm then iterates over the indices i of sequence X. For each i, it calculates the lower and upper bound of index j of sequence Y. These bounds ensure that the computation is restricted to the defined window, reducing unnecessary calculations outside the window. Within these bounds, the algorithm computes the minimum cost for each matrix cell just as the standard DTW algorithm does. Finally, the fully computed cost matrix D is returned and cell $D[N,M]$ contains the minimum alignment cost of X and Y.
The backtracking process works identically as in the standard DTW algorithm. Since the cost matrix values outside of the Sakoe-Chiba band are set to infinity, the backtracking algorithm will only consider cells within the window when tracing the optimal warp path.

Figure 3.11 shows the cost matrix of the example sequences X and Y from Figure 3.2, using a Sakoe-Chiba band with a default window size of $w = 2$. Given that X has a length of $N = 6$ and Y has a lengths of $M = 7$ the window size is calculated as $w = \max(w, \text{abs}(N-M)) = \max(2, \text{abs}(6-7)) = 2$:

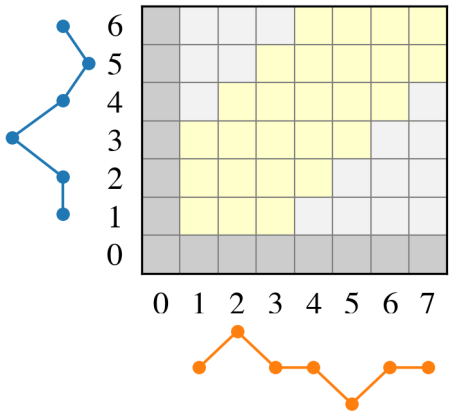

Figure 3.11: The Sakoe-Chiba band with a default window size of $w = 2$ in the cost matrix of the example sequences from Figure 3.2.

3.3 Machine Learning Models

In this section, we introduce the machine learning algorithms used in the evaluation, specifically support vector classification (SVC) and support vector regression (SVR), which are both variations of support vector machines (SVMs) [30].

3.3.1 Support Vector Classification

In support vector classification, the algorithm computes an optimal hyperplane to separate points from different classes, as shown in [30].

Definition 3.3.1 A hyperplane divides the n-dimensional feature space into two distinct half-spaces. It can be defined as the set of points $x \in \mathbb{R}^n$ that satisfy the following equation [30]:

$$w^\top x + b = 0 \tag{3.14}$$

In this equation, $w \in \mathbb{R}^n$ is the hyperplane's normal vector, which is perpendicular to the hyperplane and determines its orientation, while b specifies the distance of the hyperplane from the origin along the direction of w.

In support vector classification, a hyperplane is considered *optimal* if it maximizes the margin, which is defined as the distance between the hyperplane and the nearest data points of both classes, as described in [30]. The data points on the margin are called support vectors and the support vectors of both classes have the same distance to the hyperplane. Maximizing the margin achieves a better separation between classes, making the model less sensitive to small variations in new, unknown data [30].

To simplify the optimization problem of finding an optimal hyperplane, the decision function $w^\top x + b$ is defined such that it equals ± 1 for the support vectors, depending on their class [30]. With this normalization, maximizing the margin becomes equivalent to minimizing the norm of the hyperplane's normal vector, resulting in the following objective function [30]:

$$\min \quad \frac{1}{2}\|w\|^2 \tag{3.15}$$

The following constraint is used to ensure that data points are correctly classified [31]:

$$y_i(w^\top x_i + b) \geq 1 \tag{3.16}$$

In this equation, b specifies the distance of the hyperplane from the origin along the direction of the hyperplane's normal vector w, while x_i are the feature vectors and $y_i \in \{-1, 1\}$ are the corresponding class labels.

Given that a perfect separation of the points is not always possible, slack variables ξ_i are used to allow for some points x_i to be misclassified or fall within the margin, as shown in [30]. They quantify how much each point deviates from a correct classification. If a point is correctly classified, lying on or outside the margin, then $\xi_i = 0$. As a point moves into the margin or crosses the hyperplane (and becomes incorrectly classified), ξ_i increases. An additional regularization parameter C is used

to control the compromise between maximizing the margin and minimizing classification errors and has a default value of $C = 1$. The objective function becomes [30]:

$$\min \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \tag{3.17}$$

$C > 1$ places greater emphasis on minimizing classification errors and penalizes misclassified points more heavily. The algorithm tries to classify the training data as accurately as possible. As a result, the margin may become narrower and the model may generalize poorly to unknown data. With $C < 1$, the model penalizes misclassified points less heavily, resulting in a wider margin [30]. This can improve the model's ability to generalize, though it may result in more classification errors within the training data.

3.3.2 Support Vector Regression

"SVR [...] is an extension of support vector machines" for continuous target variables [30], [32]. The primary objective in SVR is to find a regression function $f(x) = w^\top x + b$ that predicts the output y as accurately as possible while allowing for some prediction error (within ϵ), as described in [32]. To achieve this, SVR introduces the ϵ-insensitive tube that bounds the regression function. SVR does not impose penalties on predictions falling inside this ϵ-insensitive tube. Instead, only predictions outside the ϵ-insensitive tube are penalized, proportionally to the deviation. The ϵ-insensitive tube essentially represents the region around the regression function where predictions are considered acceptable. It makes SVR more robust to small variations in the data, improving the model's ability to generalize to new data. The objective function for the optimization problem is [32]:

$$\min \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{3.18}$$

As explained in [32], the coefficient vector w defines the regression function $f(x)$ and the magnitude of w determines how strongly changes in the input x affect the predicted output $f(x)$, with a larger magnitude resulting in larger changes in the output. Minimizing the vector's norm makes the model less sensitive to small variations in the input data and it becomes more likely to generalize well to unknown data.
The parameter C controls the compromise between the width of the tube and prediction errors. The slack variables ξ_i and ξ_i^* quantify deviations of predictions outside the ϵ-insensitive tube. Specifically, ξ_i measures how much a predicted value is below the lower bound of the tube, while ξ_i^* measures how much a predicted value exceeds the upper bound [32].

The following constraints ensure that predictions either fall within the ϵ-insensitive tube or that the extent of deviation is bounded by $\epsilon + \xi_i$ and $\epsilon + \xi_i^*$. In addition, ξ_i and ξ_i^* are not negative by definition [32]:

$$\begin{aligned} y_i - (w^\top x_i + b) &\leq \epsilon + \xi_i \\ (w^\top x_i + b) - y_i &\leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* &\geq 0 \end{aligned} \tag{3.19}$$

3.4 Metrics

3.4.1 Area Under the ROC Curve

In SVC, we evaluated the models' performance using the Area Under the ROC Curve (AUC) score, which we computed with the roc_auc_score() function from the scikit-learn library [19]. The AUC ranges from 0 to 1 and reflects "a binary classification model's ability to separate positive classes from negative classes", with a perfect model achieving an AUC of 1 [33]. It measures the area under the Receiver Operating Characteristic (ROC) curve, which is a visual representation of the trade-off between the true positive rate and the false positive rate across all classification thresholds [34]. We also utilized the roc_curve() function from the scikit-learn library to compute the ROC curve [19].

3.4.2 Mean Absolute Error

In SVR, we evaluated the models' performance using the Mean Absolute Error (MAE) score, which was computed with the mean_absolute_error() function from the scikit-learn library [19]. The MAE measures the average deviation of the predicted values from the actual values, expressed in the same units as the target variable. It ranges from 0 to infinity, where a perfect model with no error attains an MAE of 0. The MAE is computed as follows, where $\hat{y}_i$ represents the predicted values, y_i represents the actual values, and n is the total number of predictions [35]:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i| \tag{3.20}$$

3.5 Classification & Regression Pipeline

Before performing the train-test split, we applied z-normalization to each time series individually, as described in [36], using the StandardScaler class of the scikit-learn library [19]. This transformation standardizes each time series to have a mean of 0 and a standard deviation of 1.
For cross-validation, we used the RepeatedStratifiedKFold and RepeatedKFold classes of the scikit-learn library, with 5 splits, 10 repeats and random_state set to 42 [19].

For classification problems, we used the SVC class of the scikit-learn library and addressed class imbalance with the `class_weight` parameter [37]. We set the `probability` parameter to `True` and used random.seed(42) from the numpy library for reproducibility [20]. Similarly, we utilized the SVR class of the scikit-learn library for regression problems [37]. We tested various C values, ranging from $C = 0.001$ to $C = 8$.
SVMs often utilize Gram matrices that store the pairwise similarities between data points [38]. We precomputed two Gram matrices (for each fold and repeat) for training and testing of the SVC and SVR models, using the normalized distance computed by the dynamic time warping algorithm as distance metric. The first Gram matrix stores the pairwise distances between the training samples and the second Gram matrix stores the pairwise distances between the test and the training samples.

In the ePOD analysis, we used SVC to predict delirium diagnosis and SVR to estimate patients' ages. Each time series feature was used individually as a predictor in both SVC and SVR. In addition, we

constructed a combined feature vector, including alpha peak, aperiodic exponent and delta power, "[z-normalizing] each each dimension [...] individually", as described in [39]. Furthermore, we repeated the analysis for the different windowing approaches.

In the MIMIC-III analysis, we used SVC to predict diabetes diagnosis with mean glucose recordings. In SVR, we used mean blood pressure and mean heart rate recordings individually to predict patient age. We repeated the analysis with both minimum length constraints.

We implemented a simple version of DTW, incorporating the Sakoe-Chiba band constraint. We tested 3 window sizes in the ePOD and MIMIC-III analysis: 5, 10 and 15.
In the ePOD analysis, we applied both normalization of the DTW distance by K and $N + M$ to assess whether the faster method could yield comparable results. In the MIMIC-III analysis, where datasets included larger patient groups, we exclusively applied normalization by $N+M$ to reduce computation time.

3.5.1 Baseline Models

For comparison, we trained simple baseline models using statistical features - specifically the mean, standard deviation, minimum and maximum values from each time series.

For each of the 8 EEG-features in the ePOD analysis, we extracted these statistical features from time series generated with fixed 4-second windows for the start of anesthesia and the end of operation.
In the MIMIC-III analysis, we included the same either 2850 or 5000 patients as in the glucose, heart rate and blood pressure data sets where recordings had a minimum length of 36. We utilized unshortened time series, instead of slicing them to a maximum length of 48, to compute the statistical values.

Before extracting the statistical features, we removed missing values from the times series. For cross-validation, we used scikit-learn's RepeatedStratifiedKFold and RepeatedKFold classes, with 5 splits, 10 repeats and random_state set to 42 [19]. We standardized the data using the StandardScaler class from scikit-learn, fitted only on the training data of each fold [19]. Next, we applied the SVC or SVR classes from scikit-learn, with a linear kernel and random_state set to 42. To address class imbalance, we used the `class_weight` parameter in SVC and tested different C values, ranging from $C = 0.001$ to $C = 8$.
We used each of the 8 EEG features and each of the 3 MIMIC-III features individually as predictors.

4 Results

4.1 ePod Results

In the following subsections, we present the AUC and MAE scores for individual features, along with the average scores across all 8 EEG features, evaluating the impact of different windowing approaches and Sakoe-Chiba band sizes on the models' predictive performance. For each combination of feature, windowing approach and Sakoe-Chiba band size, we only retained the highest AUC and lowest MAE score from multiple iterations with varying C values.

SVC Results

For the beginning of anesthesia, limiting the number of windows to 20 and 50 yielded slightly higher average AUC scores across all 8 features compared to using fixed 4-second windows, with only small differences between limiting windows to 20 versus 50. Average results were similar across band sizes. Table 4.1 presents the average AUC scores across all 8 features for different windowing approaches and band sizes at the start of anesthesia:

Time Period	**Windowing Approach**	**Mean and Standard Deviation of AUC Score across all 8 Features by Sakoe-Chiba Band Size**					
		w = 5		**w = 10**		**w = 15**	
		μ	σ	μ	σ	μ	σ
start of anesthesia	max. 20 windows	0.550	0.03	0.551	0.03	0.550	0.03
	max. 50 windows	0.553	0.02	0.548	0.02	0.550	0.04
	4-second windows	0.530	0.04	0.533	0.04	0.533	0.04

Table 4.1: Mean μ and standard deviation σ of AUC scores across all 8 EEG features for the beginning of anesthesia, using different windowing approaches and Sakoe-Chiba band sizes.

For the beginning of anesthesia, alpha power was the only feature to achieve AUC scores exceeding 0.6, consistently performing above average across all windowing approaches. We observed the highest scores using both 4-second windows and a limit of 20 windows, with a maximum score of 0.627 when using fixed windows of 4 seconds, $w = 15$ and $C = 0.001$. Table 4.2 presents the AUC scores of alpha power for the start of anesthesia, using different windowing approaches and Sakoe-Chiba band sizes.

Time Period	Feature	Windowing Approach	AUC Score by Sakoe-Chiba Band Size		
			5	10	15
start of anesthesia	alpha power	max. 20 windows	0.609	0.623	0.615
		max. 50 windows	0.574	0.591	0.604
		4-second windows	0.604	0.616	0.627

Table 4.2: AUC scores of alpha power for the beginning of anesthesia, comparing different windowing approaches and band sizes. The AUC was calculated over 10 cross-validation repeats.

The features alpha peak power and aperiodic exponent, for instance, achieved above-average results for the start of anesthesia with different window limits: alpha peak power yielded an AUC of 0.596 with a window limit of 50, using $w = 15$ and $C = 1$, while aperiodic exponent reached an AUC of 0.577 with a window limit of 20, using $w = 5$ and $C = 1$. Table 4.3 presents the results of both features for different windowing approaches and band sizes at the start of anesthesia:

Time Period	Feature	Windowing Approach	AUC Score by Sakoe-Chiba Band Size w		
			5	10	15
start of anesthesia	alpha peak power	max. 20 windows	0.510	0.521	0.520
		max. 50 windows	0.532	0.566	0.596
		4-second windows	0.511	0.528	0.511
	aperiodic exponent	max. 20 windows	0.577	0.567	0.566
		max. 50 windows	0.551	0.550	0.548
		4-second windows	0.493	0.497	0.503

Table 4.3: AUC scores of alpha peak power and aperiodic exponent for the beginning of anesthesia, comparing different windowing approaches and Sakoe-Chiba band sizes. The AUC was calculated over 10 cross-validation repeats.

The feature alpha peak also attained some above-average scores for the start of anesthesia, though in contrast to most features, it consistently reached its highest AUC scores with fixed 4-second windows. Its highest score of 0.584 was observed with a band size of $w = 5$ and $C = 0.5$. The lowest scores were mostly measured when the number of windows was limited to 20.

Table 4.4 presents the AUC scores of alpha peak for the start of anesthesia, using different windowing approaches and Sakoe-Chiba band sizes:

Time Period	**Feature**	**Windowing Approach**	**AUC Score by Sakoe-Chiba Band Size w**		
			5	**10**	**15**
start of anesthesia	alpha peak	max. 20 windows	0.529	0.524	0.529
		max. 50 windows	0.568	0.531	0.523
		4-second windows	0.584	0.576	0.571

Table 4.4: AUC scores of alpha peak for the beginning of anesthesia, comparing different windowing approaches and Sakoe-Chiba band sizes. The AUC was calculated over 10 cross-validation repeats.

In comparison with the statistical baseline models, most features (5 out of 8) showed improved performance when comparing shape similarity with DTW at the start of anesthesia.

Figure 4.1 shows the ROC curves of the best-performing feature for the beginning of anesthesia (alpha power) and its baseline model, along with the standard deviation over 10 cross-validation repeats, with fixed 4-second windows and a Sakoe-Chiba band size of $w = 15$:

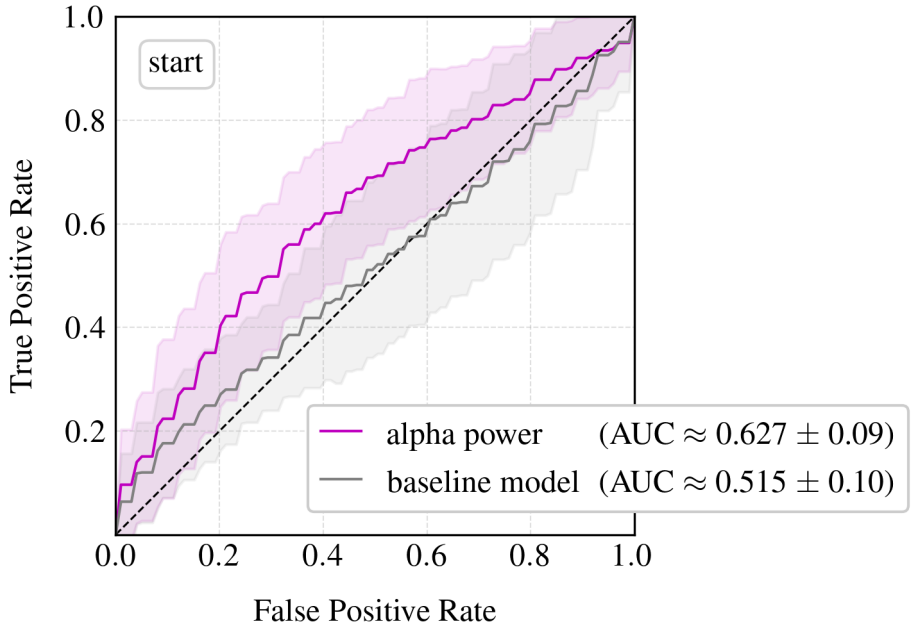

Figure 4.1: ROC curves with standard deviation over 10 cross-validation repeats of alpha power (the best-performing feature for the start of anesthesia) and its baseline model, using fixed 4-second windows and a band size of $w = 15$.

Figure 4.2 shows the ROC curves of another feature that yielded some above-average results (alpha peak) and its baseline model for the start of anesthesia, along with the standard deviation over 10 cross-validation repeats, using fixed 4-second windows and a Sakoe-Chiba band size of $w = 15$:

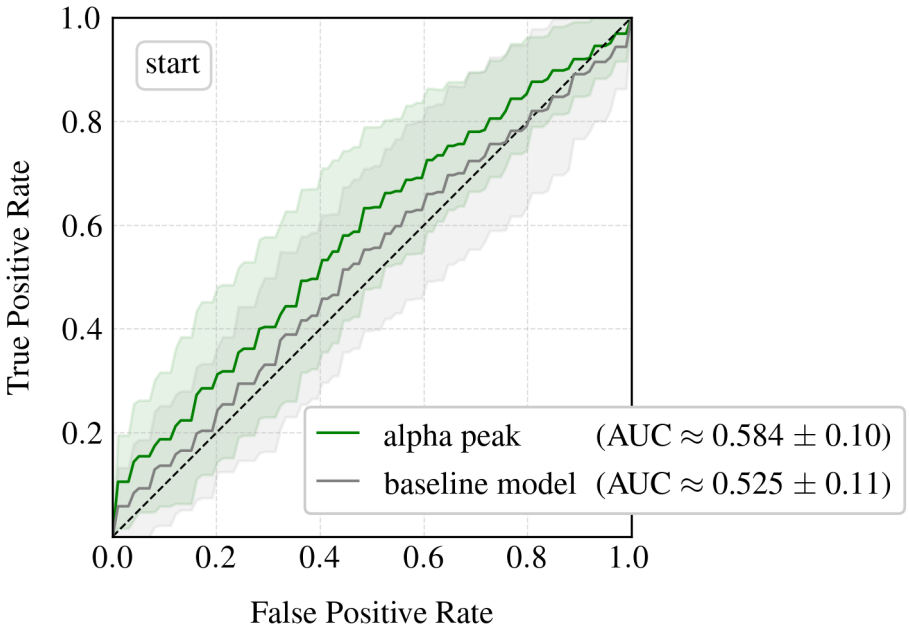

Figure 4.2: ROC curves with standard deviation over 10 cross-validation repeats of alpha peak and its baseline model for the start of anesthesia, using fixed 4-second windows and a band size of $w = 15$.

On the other hand, 3 out of 8 features, including aperiodic exponent, aperiodic offset and delta power, achieved higher AUC scores when using statistical features for the start of anesthesia.

While aperiodic exponent showed some above-average scores (0.577) with the DTW-based model, its highest AUC of 0.637 was obtained with its statistical features.
Figure 4.3 shows the ROC curve of aperiodic exponent and its baseline model for the start of anesthesia, along with the standard deviation over 10 cross-validation repeats, when limiting the number of windows to 20 and with a Sakoe-Chiba band size of $w = 5$.

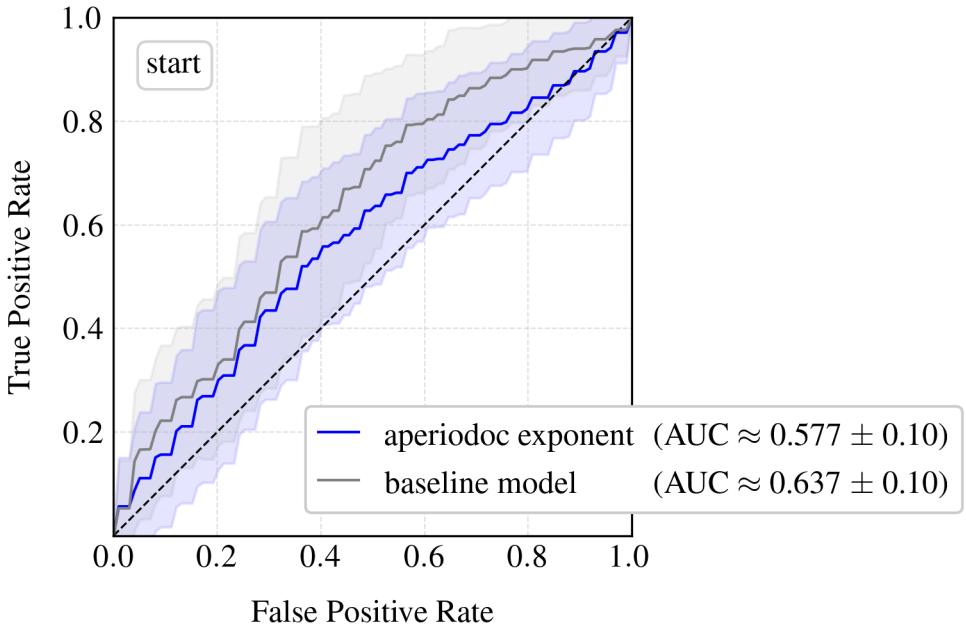

Figure 4.3: ROC curves with standard deviation over 10 cross-validation repeats of aperiodic exponent and its baseline model for the start of anesthesia, when limiting the number of windows to 20 and with a band size of $w = 5$.

In contrast, for the end of operation, the highest average AUC scores across all 8 features were consistently observed when using fixed windows of 4 seconds, with the highest average AUC score reaching 0.633. The lowest average scores were measured when the number of windows was limited to 20 and increasing the window limit to 50 yielded slightly higher scores. The results were comparable across band sizes. Table 4.5 presents the average AUC scores across all 8 features for different windowing approaches and band sizes during the end of operation:

Time Period	**Windowing Approach**	**Mean and Standard Deviation of AUC Score across all 8 Features by Sakoe-Chiba Band Size**					
		w = 5		**w = 10**		**w = 15**	
		μ	σ	μ	σ	μ	σ
end of operation	max. 20 windows	0.520	0.02	0.520	0.01	0.528	0.01
	max. 50 windows	0.534	0.04	0.543	0.04	0.542	0.04
	4-second windows	0.633	0.04	0.633	0.04	0.633	0.04

Table 4.5: Mean μ and standard deviation σ of AUC scores across all 8 EEG features for the end of operation, using different windowing approaches and Sakoe-Chiba band sizes.

Overall, the features coherence, alpha peak and aperiodic exponent achieved the highest AUC scores, using fixed 4-second windows at the end of operation. Coherence attained the highest score of 0.661, with $w = 10$ and $C = 1$. Table 4.6 presents the AUC scores of the three best-performing features for the end of operation, using fixed 4 seconds windows and different band sizes:

Time Period	**Windowing Approach**	**Feature**	**AUC Score by Sakoe-Chiba Band Size w**		
			5	**10**	**15**
end of operation	4-second windows	alpha peak	0.654	0.657	0.656
		aperiodic exponent	0.657	0.656	0.657
		coherence	0.660	0.661	0.660

Table 4.6: AUC scores of the three best-performing features for the end of operation (coherence, alpha peak and aperiodic exponent), comparing different windowing approaches and band sizes, with fixed 4-second windows. The AUC was calculated over 10 cross-validation repeats.

Compared to the statistical baseline models, *all* features from the end of operation showed improved performance when using DTW.

In contrast to coherence's highest AUC of 0.661 achieved with the DTW-based model, the baseline model attained an AUC of 0.465 for the end of operation. Figure 4.4 presents the ROC curves of coherence and its baseline model for the end of operation, along with the standard deviation over 10 cross-validation repeats, using fixed 4-second windows and a band size of $w = 10$.

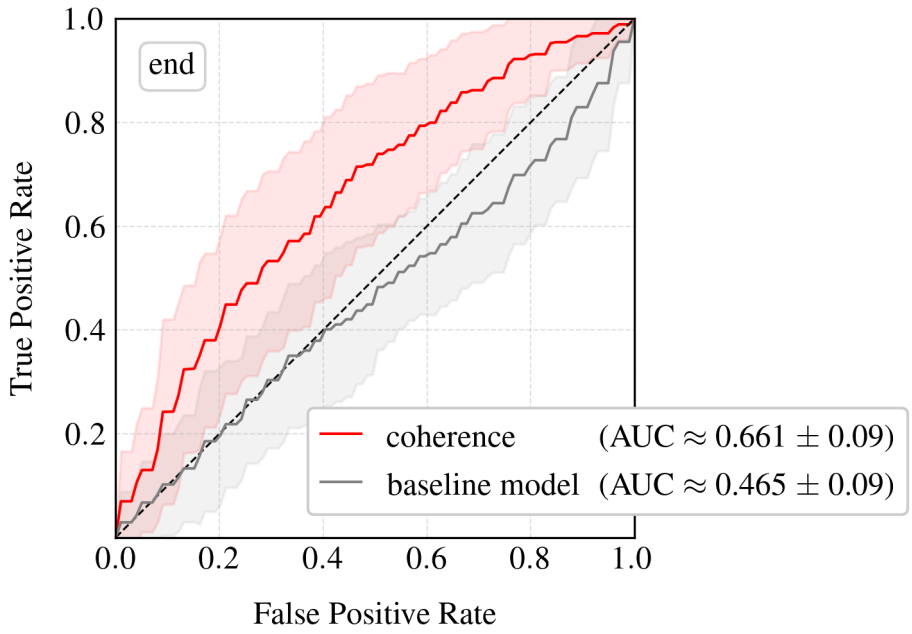

Figure 4.4: ROC curves with standard deviation over 10 cross-validation repeats of coherence and its baseline model for the end of operation, using fixed 4-second windows and a band size of $w = 10$.

Similarly, while the DTW-based model of alpha peak achieved a maximum AUC of 0.655 for the end of operation, its baseline model yielded an AUC of 0.456. Figure 4.5 shows the ROC curves of alpha peak and its baseline model for the end of operation, along with the standard deviation over 10 cross-validation repeats, using fixed 4-second windows and a Sakoe-Chiba band size of $w = 10$.

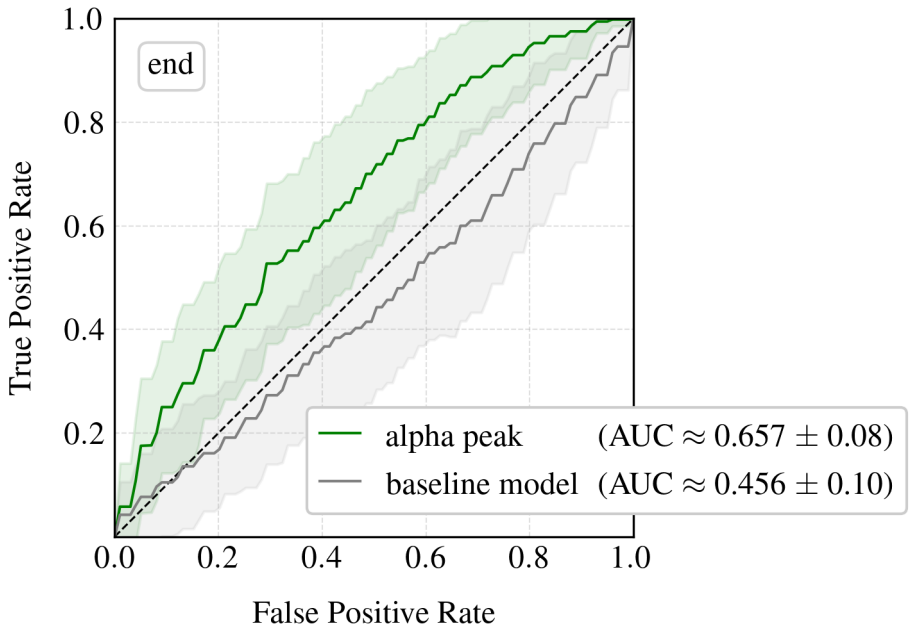

Figure 4.5: ROC curves with standard deviation over 10 cross-validation repeats of alpha peak and its baseline model for the end of operation, using fixed 4-second windows and a band size of $w = 10$.

In contrast to the start of anesthesia, aperiodic exponent's DTW-based model for the end of operation attained a slightly higher AUC of 0.657 than the corresponding baseline model, which yielded an AUC of 0.645. Figure 4.6 presents the ROC curves of aperidoc exponent and its baseline model for the end of operation, along with the standard deviation over 10 cross-validation repeats, using fixed 4-second windows and a Sakoe-Chiba band size of $w = 15$.

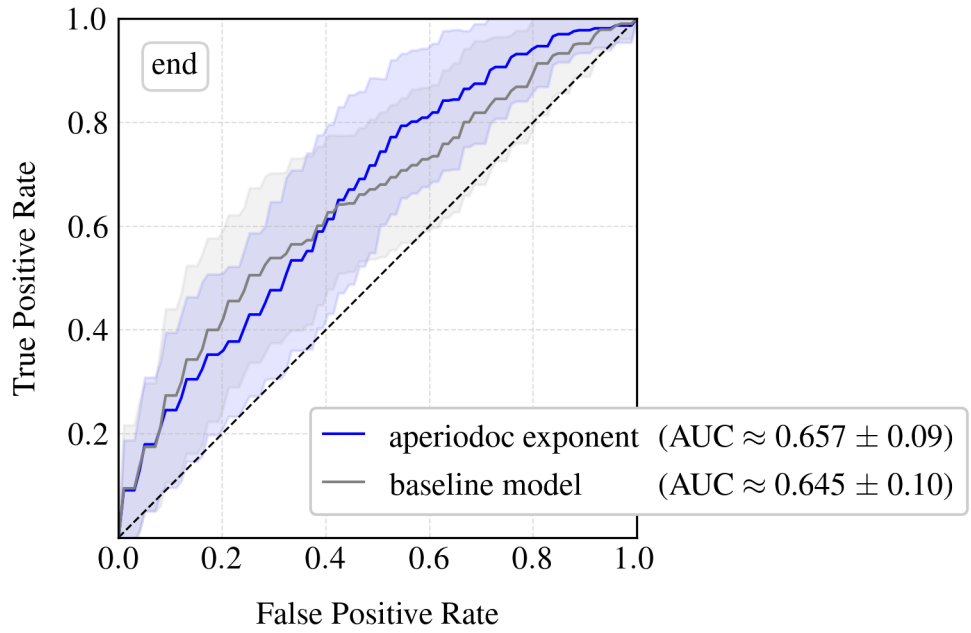

Figure 4.6: ROC curves with standard deviation over 10 cross-validation repeats of aperiodic exponent and its baseline model for the end of operation, using fixed 4-second windows and a band size of $w = 15$.

Feature Vector Results

While coherence achieved the highest AUC scores when using each feature individually as a predictor, a combined vector of alpha peak, aperiodic exponent and delta power yielded slightly higher scores, using fixed windows of 4 seconds for the end of operation, as shown in Table 4.7:

Time Period	**Windowing Approach**	**Feature Vector**	**AUC score by Sakoe-Chiba Band Size w**		
			5	**10**	**15**
end of operation	4-second windows	**[alpha peak, aperiodic exponent, delta power]**	0.667	0.667	0.669

Table 4.7: AUC scores of a feature vector combining alpha peak, aperiodic exponent and delta power for the beginning of anesthesia, comparing different windowing approaches and Sakoe-Chiba band sizes. The AUC was calculated over 10 cross-validation repeats.

SVR Results

In age prediction with SVR, we observed similar MAE scores for the start of anesthesia and the end of operation. For the start of anesthesia, the lowest average MAE across all 8 features was 4.217. For the end of the operation, the best average MAE was 4.134. Different windowing approaches affected results only slightly and outcomes were similar across band sizes.

On average, we observed slightly improved results for the baseline models. The average MAE score was 4.202 across all 8 baseline models for the start of anesthesia and 4.115 for the end of the operation.

For the start of anesthesia, restricting the number of windows to 20 consistently achieved the lowest average MAE scores across all 8 features, while fixed 4-second windows yielded the highest scores, as shown in Table 4.8:

Time Period	**Windowing Approach**	**Mean and Standard Deviation of MAE across all 8 Features by Sakoe-Chiba Band Size w**					
		w = 5		**w = 10**		**w = 15**	
		μ	σ	μ	σ	μ	σ
start of anesthesia	max. 20 windows	4.221	0.028	4.217	0.028	4.217	0.029
	max. 50 windows	4.226	0.025	4.231	0.012	4.229	0.014
	4-second windows	4.238	$2 \cdot 10^{-5}$	4.238	$2 \cdot 10^{-5}$	4.238	$4 \cdot 10^{-5}$

Table 4.8: Mean μ and standard deviation σ of MAE scores across all 8 EEG features for the start of anesthesia, using different windowing approaches and Sakoe-Chiba band sizes.

For the start of anesthesia, aperiodic exponent achieved the lowest MAE score of 4.156, by limiting the number of windows to 20 and with $w = 15$ and $C = 1$. Figure 4.7 shows the MAE scores and the standard deviation over 10 cross-validation repeats of the 3 best-performing features for the start of anesthesia, with the number of windows limited to 20 and a band size of $w = 15$ - compared to the best-performing baseline model with an MAE of 4.107:

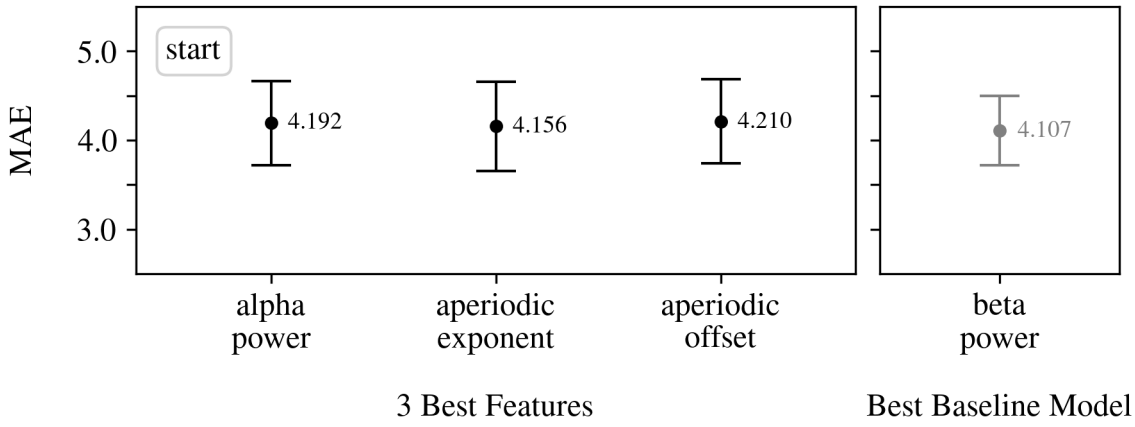

Figure 4.7: MAE and standard deviation over 10 cross-validation repeats of the 3 best-performing features (using a window limit of 20 and $w = 15$) and the best baseline model for the start of anesthesia.

For the end of operation, the lowest average MAE scores across all 8 features were consistently observed when limiting the number of windows to 50. Using fixed 4-second windows yielded slightly higher results. Table 4.9 presents the average results across all 8 features for different windowing approaches and band sizes at the end of operation:

Time Period	**Windowing Approach**	**Mean and Standard Deviation of MAE across all 8 Features by Sakoe-Chiba Band Size**					
		w = 5		**w = 10**		**w = 15**	
		μ	σ	μ	σ	μ	σ
end of operation	max. 20 windows	4.145	0.01	4.141	0.01	4.139	0.02
	max. 50 windows	4.134	0.02	4.136	0.03	4.136	0.03
	4-second windows	4.138	0.03	4.139	0.03	4.138	0.03

Table 4.9: Mean μ and standard deviation σ of MAE scores across all 8 EEG features for the end of operation, using different windowing approaches and Sakoe-Chiba band sizes.

For the end of operation, beta power achieved the lowest MAE score of 4.067 using fixed 4-second windows, $w = 5$ and $C = 1$. Figure 4.8 presents the mean MAE score and the standard deviation over 10 cross-validation repeats for the 3 best-performing features at the end of operation, using fixed 4-second windows and a band size of $w = 5$ - compared to the best-performing baseline model with an MAE of 4.006:

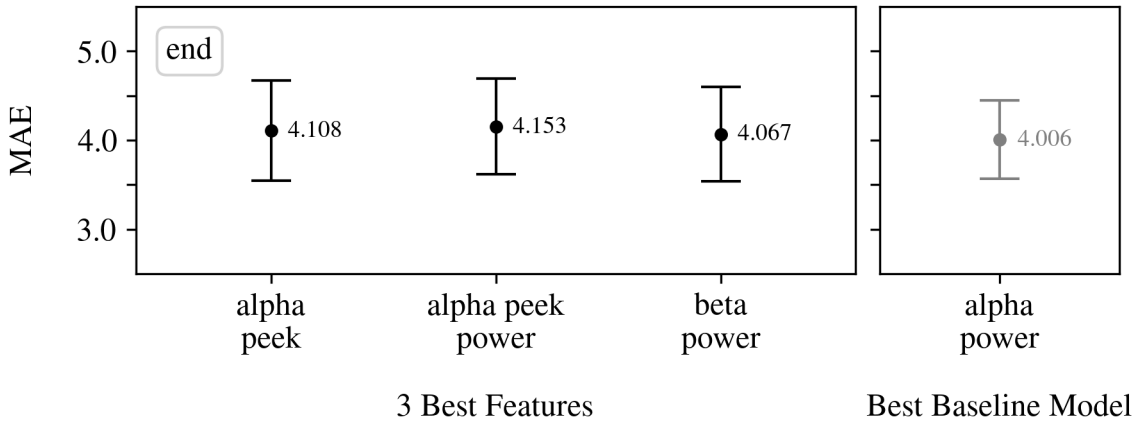

Figure 4.8: MAE and standard deviation over 10 cross-validation repeats of the 3 best-performing features (with fixed 4-second windows and $w = 5$) and the best baseline model for the end of operation.

Normalization with $N + M$ and K Results

Normalizing the DTW distance with the sum of sequence lengths $N + M$ and the warp path length K resulted in comparable average AUC and MAE scores across all 8 features, for both time periods and all windowing approaches. Table 4.10 presents the average MAE values for the end of operation, comparing normalization with $N + M$ and K, at a band size of $w = 10$.

Time Period	**Windowing Approach**	**Mean MAE Score across all 8 features by Normalization Approach**	
		N + M	**K**
end of operation	max. 20 windows	4.141	4.135
	max. 50 windows	4.136	4.130
	4-second windows	4.139	4.140

Table 4.10: Average MAE scores across all 8 EEG features during the end of operation, using a Sakoe-Chiba band size of $w = 10$. The table compares results based on windowing approach and DTW distance normalization approach.

While both normalization approaches yielded similar results, normalizing with $N + M$ reduced computation time. Table 4.11 presents the average computation time in seconds for generating a pair of Gram matrices for training and testing at the end of operation and with a band size of $w = 10$, comparing $N + M$ and K for normalization:

Time Period	**Windowing Approach**	**Mean Computation Time (in seconds) for a pair of gram matrices by Normalization Approach**	
		N + M	**K**
end of operation	max. 20 windows	1.98	8.17
	max. 50 windows	9.30	21.66
	4-second windows	282.26	326.81

Table 4.11: Average time in seconds to compute a pair of Gram matrices for training and testing for the end of operation, using a band size of $w = 10$. The table compares results based on windowing approach and DTW distance normalization approach.

4.2 MIMIC-III Results

SVC Results

In diabetes classification, we observed slightly higher AUC scores for sequences with a higher minimum length of 36. The results improved slightly with increased band size, reaching a maximum score of 0.672 at a band size of $w = 15$. Table 4.12 presents the AUC scores based on minimum sequence length and Sakoe-Chiba band size:

Minimum Length	AUC Score by Sakoe-Chiba Band Size w		
	5	**10**	**15**
24	0.647	0.662	0.663
36	0.653	0.670	0.672

Table 4.12: AUC scores for diabetes classification, comparing different minimum sequence lengths and Sakoe-Chiba band sizes. The AUC was calculated over 10 cross-validation repeats.

In comparison, the baseline model achieved an AUC of 0.791, using $C = 1$. Figure 4.9 presents the ROC curves for minimum lengths 24 and 36, with $w = 15$, alongside the baseline model's ROC curve, showing the standard deviation over 10 cross-validation repeats:

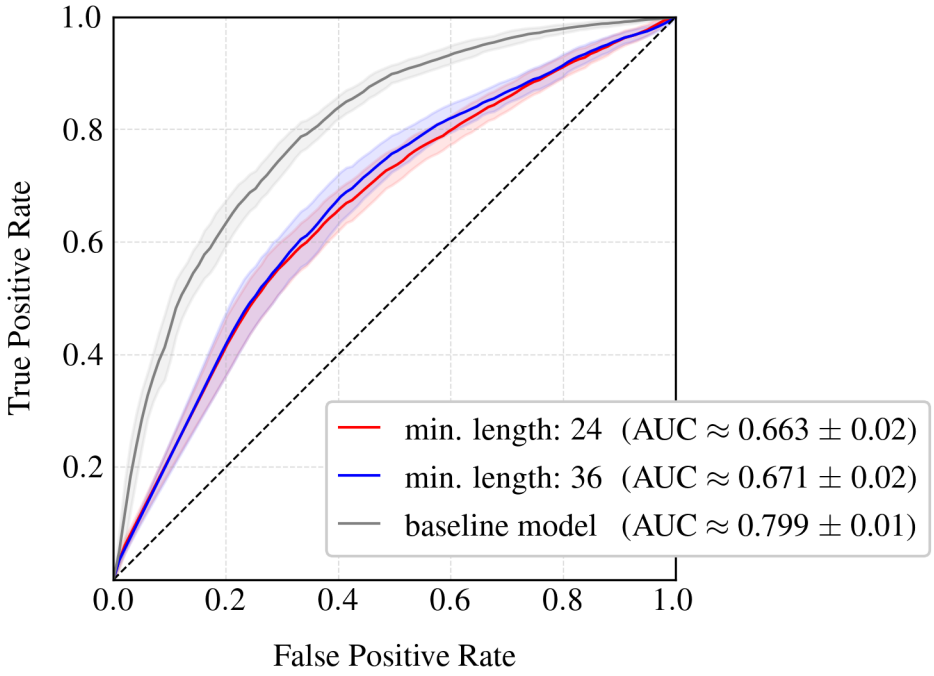

Figure 4.9: ROC curves with standard deviation over 10 cross-validation repeats, comparing minimum lengths 24 and 36 as well as the baseline model, using a band size of $w = 15$.

SVR Results

In age prediction with mean heart rate recordings, the lowest MAE score of 13.44 was obtained with sequences of minimum length 24 and with $w = 5$. Results were similar across both band sizes. Figure 4.10 presents the MAE scores with standard deviation over 10 cross-validation repeats for mean heart rate recordings, comparing minimum lengths 24 and 36, using band sizes $w = 5$ and $w = 10$:

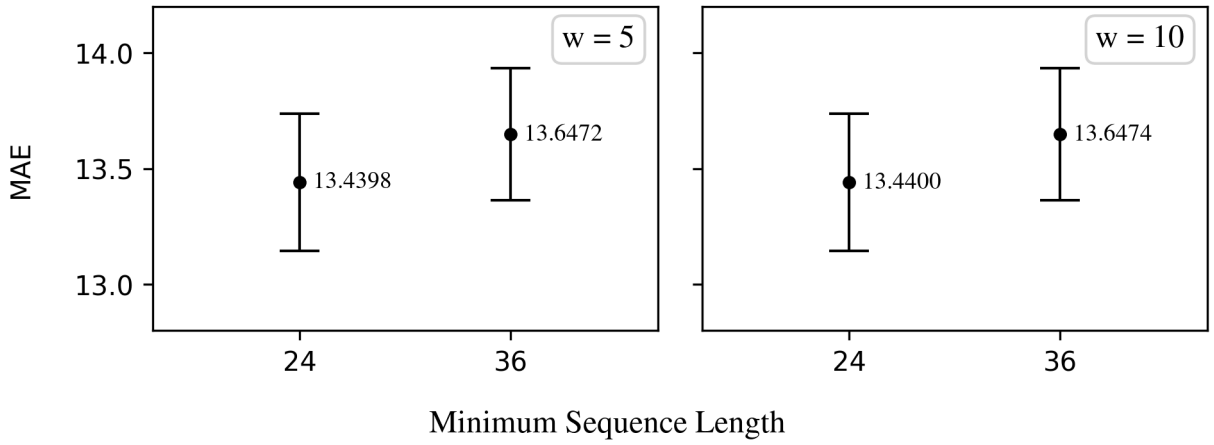

Figure 4.10: MAE scores with standard deviation over 10 cross-validation repeats in age prediction using mean heart rate recordings, comparing different minimum lengths and band sizes.

We observed similar results in age prediction with mean blood pressure recordings, yielding a slightly lower minimum MAE of 13.25, using $w = 5$. The results were again similar across band sizes. In contrast, sequences with a minimum length of 36 achieved slightly improved MAE scores. Figure 4.11 presents the MAE scores with standard deviation over 10 cross-validation repeats for mean blood pressure recordings, comparing minimum lengths 24 and 36, using band sizes $w = 5$ and $w = 10$:

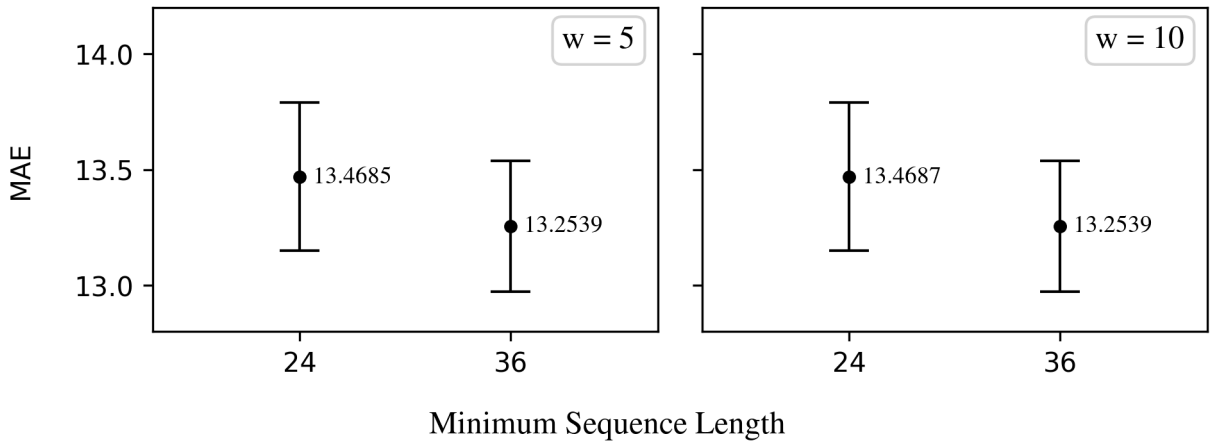

Figure 4.11: MAE scores with standard deviation in age prediction using mean blood pressure recordings, comparing different minimum lengths and band sizes.

In comparison, both baseline models produced considerably higher MAE scores. The baseline model for heart rate time series yielded an MAE of 24.99 and the baseline model for blood pressure time series attained an MAE of 25.45. Table 4.13 presents a comparison between the results of the DTW-based models for both features (with minimum length 36 a band size of $w = 5$) and the corresponding baseline models:

Feature	**MAE Score by Model**			
	DTW Model		**Baseline Model**	
	μ	σ	μ	σ
heart rate	13.647	0.286	24.986	1.418
blood pressure	13.254	0.286	25.449	1.488

Table 4.13: MAE scores with standard deviation over 10 cross-validation repeats for heart rate and blood pressure time series. The table compares the results of the DTW-based models (with minimum length 36 and $w = 5$) against the baseline models.

5 Discussion

5.1 Discussion of ePod Results

For the start of anesthesia, alpha power was the only feature to yield AUC scores exceeding 0.6 for delirium classification, with a maximum score of 0.63. It consistently achieved above-average results across different Sakoe-Chiba band sizes and windowing approaches, while the corresponding baseline model yielded a considerably lower AUC of 0.515. This may suggest that the shape of alpha power time series from the start of anesthesia provides some predictive value for delirium classification.
While most features performed best when the number of windows was restricted to 20 or 50 for the start of anesthesia, AUC scores for these window-limiting approaches averaged around 0.55, which is just better than random guessing. The best-performing feature, alpha power, yielded similar results across all windowing approaches. In addition, other features that attained above-average scores did so with varying windowing approaches, indicating that no single windowing approach consistently yields better results for the start of anesthesia.
Some features, specifically aperiodic exponent, aperiodic offset and delta power, showed improved performance with their corresponding statistical baseline models, suggesting that DTW may only be of limited use and effectiveness for these features at the start of anesthesia - particularly given its higher time complexity. Although aperiodic exponent yielded some above-average results with DTW, it achieved the highest AUC of 0.64 attained with statistical features for the start of anesthesia.
The generally lower predictive performance observed at the start of anesthesia, compared to the end of operation, further indicates that features derived from recordings of this time period may not contain sufficient similarities in shape related to an onset of delirium needed for reliable delirium prediction with DTW - with alpha power standing out as a relative exception.

Overall, we observed notably higher AUC scores for the end of operation, averaging 0.63 across all 8 features when using fixed 4-second windows. This outcome suggests that features derived from EEG recordings of this time period may contain more distinct similarities in shape associated with an onset of delirium.
Considering that all features, including aperiodic exponent, yielded improved results with DTW compared to their corresponding statistical baseline models, DTW appears to be an effective distance metric for delirium classification when using recordings from the end of the operation.
Coherence achieved the overall highest AUC of 0.66, which indicates that the model has a reasonable chance of ranking a positive instance above a negative one.
Fixed windows of 4 seconds consistently yielded the highest results for the end of operation, showing an advantage over limiting window numbers and suggesting that restricting the number of windows may not be well-suited for preserving patterns in shape associated with delirium.

The slightly improved AUC results observed when using a feature vector combining alpha peak, aperiodic exponent and delta power suggest that incorporating more information may provide a more

suitable representation required for predicting an onset of delirium. Meanwhile, calculating the Euclidean distance between a pair of vectors requires only $O(d)$ time, where d is the dimensionality of the vectors. This is a relatively minor computational addition compared to the overall complexity of DTW, which is $O(N \times M)$.

The comparable MAE values observed in age prediction, with an average of 4.23 for the beginning of anesthesia and 4.14 for the end of the operation, suggest that the EEG features may contain comparable, age-related predictive information for elderly patients regardless of the operation phase.
Given the patients' age range of 70 to 92, spanning 23 years, an MAE of around 4 years indicates that the models capture some age-related distinctions in the shape of the features, though precision remains moderate. This suggests that the extracted features may capture only limited information about shape similarities related to age.
Considering the slightly lower average MAE scores of the baseline models and DTW's higher time complexity, DTW may only be of limited use and effectiveness as a metric for age prediction with EEG recordings. However, since the baseline model's MAE results are still comparable and similarly moderate, age prediction also appears challenging with statistical features from EEG recordings.

Furthermore, results indicate that normalizing the DTW distance by the sum of the sequence lengths $N + M$ is an effective approach for analyzing EEG time series features, achieving results similar to normalization by the warp path length K while improving model efficiency.
Adjusting the Sakoe-Chiba band size yielded varying effects across different features and windowing approaches. Overall, the impact of band size on results was minimal, though setting $w = 5$ provided a modest reduction in computation time.

Despite the application of z-normalization to standardize the data, noise may have contributed to the rather modest AUC and MAE scores observed in this analysis, particularly for the start of anesthesia. EEG recordings are susceptible to noise, which can mask subtle patterns in the brain's electrical activity, thereby affecting accurate prediction of delirium onset and age, posing a potential limitation to this thesis [40]. Given the inherently noisy nature of the data, AUC scores of 0.66 and 0.69 may be considered reasonably respectable.

Both this thesis and Shi et al.'s study focused on DTW as a distance metric in classification of EEG features, though within different contexts and with different objectives. While Shi et al. classified EEG recordings to identify imagined movements, our analysis utilized features extracted from EEG recordings collected during operations on elderly patients, aiming to predict an onset of delirium. We implemented a rather simple version of DTW and used SVC with 10 cross-validation repeats, yielding an AUC of 0.69. In contrast, Shi et al. used a combination of Wavelet Packet Decomposition and DTW for enhanced feature extraction and applied Quadratic Discriminant Analysis, achieving an accuracy of 90.89% and making DTW an effective distance metric for classification in this context [7].

5.2 Discussion of MIMIC-III Results

In diabetes classification, a maximum AUC score of 0.67 indicates that shape similarity of mean glucose recordings may provide some predictive information, suggesting that the model has a reasonable

chance of ranking a positive instance above a negative one, though predictive accuracy remains moderate.
Time series with a maximum length of 36, which have a smaller standard deviation in length, achieved slightly higher AUC scores, suggesting that reduced variation in sequence length may contribute to slightly improved AUC scores in diabetes classification. The slight differences in results for varying minimum lengths may also suggest that normalization by $N+M$, intended to account for variations in the length of sequences, does not fully balance the effect of length variation on the DTW distance in this context. Additionally, time series in the dataset with a minimum length of 36 have a higher mean length, capturing more information over a longer time period, which may have slightly improved predictive accuracy. Besides, since each dataset had a different minimum length requirement for the time series, they included slightly different groups of patients. This difference in the data may also explain small variations in the results.
Increasing the Sakoe-Chiba band size consistently resulted in slightly higher AUC scores in diabetes classification. A larger band size enables the algorithm to account for greater shifts in timing between samples, suggesting that these time series may exhibit more substantial timing variation, and more flexibility in aligning the sequences may have improved the model's ability to capture similarities that occur at different times across samples. In addition, we did not apply any domain-specific markers to define the start and end points of extracted MIMIC-III time series and instead applied general minimum and maximum length constraints. This may have also necessitated a larger band size to account for the variability introduced by missing domain-specific markers.
In contrast, the simple baseline model not only achieved a considerably higher AUC score of 0.79 but also reduced computation time substantially, suggesting that statistical feature extraction is a more effective and efficient strategy. Thus, applying DTW to classify hourly aggregated glucose time series from MIMIC-Extract may not be a well-suited approach to predict diabetes.
The inclusion of patients with undiagnosed diabetes in the dataset could have also affected the results of diabetes classification. According to data from the CDC, approximately 22.8% of U.S. adults with diabetes were unaware of their condition in 2021 [41]. These undiagnosed patients would have been included as non-diabetic instances, potentially affecting the model's interpretation of shape patterns in glucose recordings and reducing classification accuracy - posing an additional limitation to this thesis. However, this should have similarly affected the baseline model, which nonetheless achieved a considerably higher AUC.
The extracted glucose time series contained a substantial number of missing entries, averaging 70 or 72 in each time series. While removing missing entries does not distort the sequential order of valid entries, their occurrence effectively extends the time period represented by the extracted time series. Consequently, the density of information is relatively low over this stretched time period, which may have affected shape similarity comparisons with DTW, presenting another limitation of this thesis.

In age prediction, we observed a minimum MAE of 13.44 years with heart rate time series and 13.25 years with blood pressure time series. Given the age range of 15 to 89 (spanning 75 years), this MAE reflects some ability to capture general age-related similarities in heart rate and blood pressure recordings. However, the relatively high error indicates that age prediction from similarities of hourly aggregated heart rate or blood pressure recordings alone may be challenging.
When utilizing heart rate time series, shorter sequences with a minimum length of 24 yielded a lower MAE, while blood pressure time series required longer sequences with a minimum length of 36 to achieve an improved MAE. These findings may suggest that adjusting the minimum sequence length

according to the specific medical parameter and its biological rhythms affects model accuracy.
In comparison with the baseline models, which yielded an MAE of 24.99, shape similarity of hourly aggregated heart rate and blood pressure time series appears to provide some, though limited, predictive value. Given the overall moderate MAE results from both the DTW-based models and the baseline models of MIMIC-III features, accurate age prediction may be generally challenging in this context.
In addition, the general medical condition of ICU patients may have influenced the moderate MAE results obtained from MIMIC-III features. Patients treated in ICUs typically experience critical health issues, which can significantly affect physiological parameters such as blood pressure and heart rate, making them less reflective of age-related values.

Furthermore, MIMIC-Extract aggregated time series into hourly intervals, potentially limiting the amount of information preserved [17]. This hourly aggregation may have concealed dynamic variations within the time series that could have been important for capturing meaningful shape information and achieving accurate diabetes classification and age prediction. If these time series shapes only contain little predictive information to begin with, this inherently constrains our ability to assess the effectiveness of DTW for capturing meaningful similarities. Consequently, using the aggregated time series from MIMIC-Extract may not have been an appropriate choice, posing a considerable limitation of this thesis.
Additionally, missing domain knowledge may have impacted choices important to the analysis, such as selecting relevant features, handling of missing values or defining domain-specific markers for time series extraction from the MIMIC-III data.
While normalization by $N+M$ yielded comparable results to normalization by K for the EEG features, this may not have been true for the MIMIC-III features. However, since we only utilized normalization by $N + M$ for MIMIC-III features, we did not evaluate the effect of this normalization method compared to alternatives, such as normalization by K.

In contrast to Huang and Kinsner's 2002 study, which used recordings of the heart's electrical activity, we utilized heart rate recordings in this thesis. Given the hourly aggregation by MIMIC-Extract, we focused on the hourly progression of heart rate, rather than examining the heart's electrical activity at the level of individual heartbeats. Furthermore, while Huang and Kinsner considered the specific waveform structure, we did not apply any domain-specific markers to define the start and end of the time series. We also implemented a relatively simple version of DTW, whereas Huang and Kinsner adjusted the alignment cost computation by evaluating similarity using the mean of absolute residuals along the alignment path. Finally, while Huang and Kinsner used ECG recordings for classification tasks, we utilized heart rate recordings for regression tasks to predict patient age, which limits the direct comparability of results [6].

6 Conclusion

In this thesis, we further evaluated DTW as a distance metric for classification of medical time series data, using SVC and SVR. We utilized 8 time series features derived from perioperative EEG recordings of elderly patients, aiming to predict delirium diagnosis and patient age. In addition, we utilized hourly aggregated glucose, heart rate, and blood pressure time series from the MIMIC-III database, aiming to predict diabetes diagnosis and patient age.
In delirium classification, we compared two distinct operative periods and observed notably higher AUC scores for the end of operation than for the start of anesthesia - with alpha power standing out as a relative exception, achieving an AUC of 0.63 for the start of anesthesia. Coherence was the best-performing feature, achieving an AUC of 0.66 for the end of operation, and a combined feature vector including alpha peak, aperiodic exponent and delta power yielded an AUC of 0.69. For the end of operation, DTW-based models consistently attained higher AUC scores than the corresponding statistical baseline models. We examined different windowing approaches and found that fixed 4-second windows yielded the highest AUC scores. In age prediction with EEG features, we observed a minimum MAE of 4.07 years across an age range of 23 years. Normalization by the sum of the sequence lengths achieved comparable results to normalization by the warp path length, while also reducing computation time.
In diabetes classification, we attained a maximum AUC of 0.67, while a much faster baseline model, which relied solely on statistical features, yielded an AUC of 0.79. In age prediction with heart rate and blood pressure time series, we observed a minimum MAE of 13.25, across an age range of 75 years.
In conclusion, DTW proved to be a relatively effective distance metric for predicting an onset of delirium, when using features derived from the end of operation and fixed 4-second windows. Notably, DTW also appeared to be effective for alpha power time series derived from the start of anesthesia. Given the inherently noisy nature of EEG data, AUC scores of 0.66 and 0.69 may be considered reasonably respectable. The improved results from using fixed 4-second windows suggested that preserving more detailed information is important for accurate delirium prediction. Additionally, normalization with the sum of the sequence lengths presented a practical optimization for the analysis of perioperative EEG features with DTW. The moderate MAE scores indicated that age prediction from similarities of perioperative EEG time series and hourly aggregated time series is challenging. We also determined that the hourly aggregation of MIMIC-III time series through MIMIC-Extract presented a substantial limitation to this thesis and that DTW may not be a well-suited approach in this context.
Some unanswered questions remain open for future research. In the context of these specific datasets, it may be worth investigating whether more detailed glucose, heart rate, and blood pressure time series could improve diabetes and age prediction. In addition, analyzing different approaches to normalizing the DTW distance for MIMIC-III features could enhance the evaluation of DTW as a distance metric in medical data analysis. Finally, utilizing other classification techniques besides SVC and SVR would contribute to the analysis of DTW as a distance metric for classification in medical data analy-

sis. In particular, the 1-nearest neighbor approach, highlighted as a potentially effective technique in numerous studies, could be applied to the ePod and MIMIC-III features [42], [43].

Overall, we observed varied results depending on the chosen feature, the level of aggregation and the prediction objective. While DTW was a relatively effective distance metric for delirium classification with EEG features from the end of the operation, it proved less suitable for diabetes classification and age prediction with hourly aggregated physiological measurements from MIMIC-Extract.

AI Tools

ChatGPT

ChatGPT (GPT-4o model), developed by OpenAI, was used to help in summarizing relevant papers, though we didn't copy any text generated by ChatGPT.
`https://chatgpt.com/`

LanguageTool

LanguageTool (free version) was used to correct spelling for English and German throughout parts of the thesis.
`https://languagetool.org/`

Grammarly

Grammarly (free version), developed by Grammarly Inc., was used to correct grammar and spelling throughout the entire thesis.
`https://app.grammarly.com/`

Bibliography

[1] H. Kamper, "Dynamic time warping," accessed: 2024-11-18. [Online]. Available: https://www.kamperh.com/slides/dtw-crop.pdf

[2] S. Dash, S. K. Shakyawar, M. Sharma, and S. Kaushik, "Big data in healthcare: management, analysis and future prospects," *Journal of big data*, vol. 6, no. 1, pp. 1–25, 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0217-0

[3] A. E. Johnson, T. J. Pollard, L. Shen, L.-w. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, and R. G. Mark, "Mimic-iii, a freely accessible critical care database," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016. [Online]. Available: https://doi.org/10.1038/sdata.2016.35

[4] V. Tuzcu and S. Nas, "Dynamic time warping as a novel tool in pattern recognition of ecg changes in heart rhythm disturbances," in *2005 IEEE international conference on systems, man and cybernetics*, vol. 1. IEEE, 2005, pp. 182–186. [Online]. Available: https://ieeexplore.ieee.org/document/1571142

[5] P. C. Ivanov, L. A. N. Amaral, A. L. Goldberger, S. Havlin, M. G. Rosenblum, Z. R. Struzik, and H. E. Stanley, "Multifractality in human heartbeat dynamics," *Nature*, vol. 399, no. 6735, pp. 461–465, 1999. [Online]. Available: https://doi.org/10.1038/20924

[6] B. Huang and W. Kinsner, "Ecg frame classification using dynamic time warping," in *IEEE CCECE2002. Canadian Conference on Electrical and Computer Engineering. Conference Proceedings (Cat. No. 02CH37373)*, vol. 2. IEEE, 2002, pp. 1105–1110. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1013101

[7] Y. Shi, F. Li, T. Liu, F. R. Beyette, and W. Song, "Dynamic time-frequency feature extraction for brain activity recognition," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2018, pp. 3104–3107. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8512914

[8] H.-C. Huang and B. Jansen, "Eeg waveform analysis by means of dynamic time-warping," *International journal of bio-medical computing*, vol. 17, no. 2, pp. 135–144, 1985. [Online]. Available: https://doi.org/10.1016/0020-7101(85)90084-4

[9] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE transactions on acoustics, speech, and signal processing*, vol. 26, no. 1, pp. 43–49, 1978. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1163055

[10] G. A. Ten Holt, M. J. Reinders, and E. A. Hendriks, "Multi-dimensional dynamic time warping for gesture recognition," in *Thirteenth annual conference of the Advanced*

School for Computing and Imaging, vol. 300, 2007, p. 1. [Online]. Available: https://www.researchgate.net/publication/228740947

[11] J. Aach and G. M. Church, "Aligning gene expression time series with time warping algorithms," *Bioinformatics*, vol. 17, no. 6, pp. 495–508, 2001. [Online]. Available: https://doi.org/10.1093/bioinformatics/17.6.495

[12] K. LaFleur, K. Cassady, A. Doud, K. Shades, E. Rogin, and B. He, "Quadcopter control in three-dimensional space using a noninvasive motor imagery-based brain–computer interface," *Journal of neural engineering*, vol. 10, no. 4, p. 046003, 2013. [Online]. Available: https://iopscience.iop.org/article/10.1088/1741-2560/10/4/046003/pdf

[13] M. Pollak, S. Leroy, V. Röhr, E. N. Brown, C. Spies, and S. Koch, "Electroencephalogram biomarkers from anesthesia induction to identify vulnerable patients at risk for postoperative delirium," *Anesthesiology*, vol. 140, no. 5, pp. 979–989, 2024. [Online]. Available: https://doi.org/10.1097/ALN.0000000000004929

[14] E. Larson, A. Gramfort, D. A. Engemann, J. Leppakangas, C. Brodbeck, M. Jas, T. L. Brooks, J. Sassenhagen, D. McCloy, M. Luessi, J.-R. King, R. Höchenberger, R. Goj, G. Favelier, C. Brunner, M. van Vliet, M. Wronkiewicz, A. Rockhill, C. Holdgraf, and luzpaz, "Mne-python (v1.8.0)," 2024. [Online]. Available: https://doi.org/10.5281/zenodo.13340330

[15] T. Donoghue, M. Haller, E. J. Peterson, P. Varma, P. Sebastian, R. Gao, T. Noto, A. H. Lara, J. D. Wallis, R. T. Knight *et al.*, "Parameterizing neural power spectra into periodic and aperiodic components," *Nature neuroscience*, vol. 23, no. 12, pp. 1655–1665, 2020. [Online]. Available: https://doi.org/10.1038/s41593-020-00744-x

[16] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright *et al.*, "Scipy 1.0: fundamental algorithms for scientific computing in python," *Nature methods*, vol. 17, no. 3, pp. 261–272, 2020. [Online]. Available: https://www.nature.com/articles/s41592-019-0686-2

[17] S. Wang, M. B. McDermott, G. Chauhan, M. Ghassemi, M. C. Hughes, and T. Naumann, "Mimic-extract: A data extraction, preprocessing, and representation pipeline for mimic-iii," in *Proceedings of the ACM conference on health, inference, and learning*, 2020, pp. 222–235. [Online]. Available: https://doi.org/10.1145/3368555.3384469

[18] A. Amaize and K. B. Mistry, *Emergency Department Visits for Children and Young Adults With Diabetes, 2012*, ser. Healthcare Cost and Utilization Project (HCUP) Statistical Briefs. Rockville, MD: Agency for Healthcare Research and Quality (US), 2016. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK368403/

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011. [Online]. Available: https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf

[20] C. R. Harris, K. J. Millman, S. J. Van Der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith *et al.*, "Array programming with numpy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[21] S. R. Eddy, "What is dynamic programming?" *Nature biotechnology*, vol. 22, no. 7, pp. 909–910, 2004. [Online]. Available: https://doi.org/10.1038/nbt0704-909

[22] R. Herzog, "Lecture notes: Introduction to optimization," 2023, accessed: 2024-11-18. [Online]. Available: https://scoop.iwr.uni-heidelberg.de/teaching/2022ws/short-course-optimization/introduction-to-optimization-lecture-notes-20230329.pdf

[23] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 561–580, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306437915000733

[24] L. Wang, Y. Zhang, and J. Feng, "On the euclidean distance of images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1334–1339, 2005.

[25] S. Fong, "Using hierarchical time series clustering algorithm and wavelet classifier for biometric voice classification," *BioMed Research International*, vol. 2012, no. 1, p. 215019, 2012. [Online]. Available: https://doi.org/10.1155/2012/215019

[26] P. Tormene, T. Giorgino, S. Quaglini, and M. Stefanelli, "Matching incomplete time series with dynamic time warping: an algorithm and an application to post-stroke rehabilitation," *Artificial intelligence in medicine*, vol. 45, no. 1, pp. 11–34, 2009. [Online]. Available: https://doi.org/10.1016/j.artmed.2008.11.007

[27] E. J. Keogh and M. J. Pazzani, "Scaling up dynamic time warping to massive datasets," in *Principles of Data Mining and Knowledge Discovery: Third European Conference, PKDD'99, Prague, Czech Republic, September 15-18, 1999. Proceedings 3*. Springer, 1999, pp. 1–11. [Online]. Available: https://doi.org/10.1007/978-3-540-48247-5_1

[28] P. Senin, "Dynamic time warping algorithm review," Information and Computer Science Department, University of Hawaii at Manoa, Technical Report 08-04, 2008, accessed: 2024-11-18. [Online]. Available: https://csdl.ics.hawaii.edu/techreports/2008/08-04/08-04.pdf

[29] S. Bae and S. Bae, "Big-o notation," *JavaScript Data Structures and Algorithms: An Introduction to Understanding and Implementing Core Data Structure and Algorithm Fundamentals*, pp. 1–11, 2019. [Online]. Available: https://doi.org/10.1007/978-1-4842-3988-9_1

[30] R. G. Brereton and G. R. Lloyd, "Support vector machines for classification and regression," *Analyst*, vol. 135, no. 2, pp. 230–267, 2010. [Online]. Available: https://pubs.rsc.org/en/content/articlelanding/2010/an/b918972f

[31] A. Ng, "Cs229 lecture notes part v support vector machines," accessed: 2024-11-18. [Online]. Available: https://see.stanford.edu/materials/aimlcs229/cs229-notes3.pdf

[32] M. Awad, R. Khanna, M. Awad, and R. Khanna, "Support vector regression," *Efficient learning machines: Theories, concepts, and applications for engineers and system designers*, pp. 67–80, 2015. [Online]. Available: https://doi.org/10.1007/978-1-4302-5990-9_4

[33] G. Developers, "Machine learning glossary auc (area under the roc curve)," accessed: 2024-11-18. [Online]. Available: https://developers.google.com/machine-learning/glossary#AUC

[34] Google Developers, "Machine learning crash course classification: Roc and auc," accessed: 2024-11-18. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

[35] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate research*, vol. 30, no. 1, pp. 79–82, 2005. [Online]. Available: https://www.int-res.com/articles/cr2005/30/c030p079.pdf

[36] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 262–270. [Online]. Available: https://doi.org/10.1145/2339530.2339576

[37] scikit-learn Developers, "Support Vector Machines," accessed: 2024-11-17. [Online]. Available: https://scikit-learn.org/stable/modules/svm.html

[38] C. Cortes, "Support-vector networks," *Machine Learning*, 1995. [Online]. Available: https://ise.ncsu.edu/wp-content/uploads/sites/9/2022/08/Cortes-Vapnik1995_Article_Support-vectorNetworks.pdf

[39] M. Łuczak, "Combining raw and normalized data in multivariate time series classification with dynamic time warping," *Journal of Intelligent & Fuzzy Systems*, vol. 34, no. 1, pp. 373–380, 2018. [Online]. Available: https://www.researchgate.net/publication/322609711

[40] S. L. Kappel, D. Looney, D. P. Mandic, and P. Kidmose, "Physiological artifacts in scalp eeg and ear-eeg," *Biomedical engineering online*, vol. 16, pp. 1–16, 2017.

[41] C. for Disease Control and Prevention, "Data & research on diabetes," 2021, accessed: 2024-11-05. [Online]. Available: https://www.cdc.gov/diabetes/php/data-research/index.html

[42] X. Xi, E. Keogh, C. Shelton, L. Wei, and C. A. Ratanamahatana, "Fast time series classification using numerosity reduction," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 1033–1040. [Online]. Available: https://doi.org/10.1145/1143844.1143974

[43] Z. Xing, J. Pei, and E. Keogh, "A brief survey on sequence classification," *ACM Sigkdd Explorations Newsletter*, vol. 12, no. 1, pp. 40–48, 2010. [Online]. Available: https://doi.org/10.1145/1882471.1882478