ulm university universität
uulm

Universität Ulm | 89069 Ulm | Germany

Faculty of Engineering, Computer Science and Psychology
Institute of Neural Information Processing

Methods of Anomaly Detection for Pain Recognition in Biophysiological Time Series

Bachelor thesis at Ulm University.

Submitted for the degree of Bachelor of Science (BSc) in Computer Science.

Submitted by:

Yannick Exner

yannick.exner@uni-ulm.de

Matriculation number: 1066556

Thesis advisor:

Prof. Dr. Friedhelm Schwenker

Ulm University, Ulm

Ulm, December 2024

Version December 03, 2024

Name: Yannick Exner Matriculation number: 1066556

Declaration

I hereby declare that I wrote the bachelor thesis independently and used no other aids than those cited. In each individual case, I have clearly identified the source of the passages that are taken word for word or paraphrased from other works.

Ulm, 03 Dec 2024 Yannick Exner

Yannick Exner

Abstract

Traditional pain assessment methods rely on subjective self-report, which may be impractical or inaccurate in various scenarios. Consequently, methods for objective and automated pain assessment have become an active area of research. This study evaluates unsupervised anomaly detection (AD) methods for pain recognition in biophysiological time series using data from the BioVid Heat Pain Database (BVDB) Part A. Unlike conventional supervised classification approaches, the AD methods explored here are trained exclusively on normal (no pain) data. Three widely used AD methods – Matrix Profile, Isolation Forest, and Autoencoder – are evaluated for their effectiveness in detecting pain-related anomalies in electrodermal activity (EDA), electrocardiogram (ECG), and electromyogram (EMG) signals. These AD methods are also compared against simple baseline AD approaches, PolyFit and ArgMax, and a supervised random forest classifier.

The results reveal that EDA is the simplest and most informative signal, achieving an area under the curve (AUC) of 88.05% with Isolation Forest and 87.92% with PolyFit. Interestingly, the simplest approach, ArgMax, which uses the position of the maximum value in the EDA signal as the anomaly score, achieved the highest accuracy of 82.61%. ECG and EMG signals were less effective individually, with their best accuracies reaching 72.62% and 58.23%, respectively, using Matrix Profile. These results suggest that AD methods offer a viable alternative to supervised classification methods for pain detection in scenarios where labeled data is unavailable or when the detection of novel anomalies beyond pain-related events is of interest.

Contents

1 Introduction

Pain is a complex and subjective experience that serves as a critical indicator of potential or actual harm to the body. Traditional pain assessment methods predominantly rely on subjective self-reporting, such as the Numerical Rating Scale (NRS), where patients rate their pain on a scale from 0 to 10, or the Visual Analog Scale (VAS), where patients mark their pain level on a 10 cm line [1]. While practical, these methods have significant limitations, including their inherent subjectivity, limited applicability for non-communicative patients, and inability to provide continuous monitoring. Consequently, substantial research efforts have focused on developing objective and automated pain assessment methods using biophysiological signals or video data [2, 3, 4].

The BioVid Heat Pain Database (BVDB) [5] has been extensively used in classification tasks for nociceptive heat-induced pain recognition. BVDB is a multimodal dataset that contains frontal video and biomedical signals, including electrodermal activity (EDA), electrocardiogram (ECG) and electromyogram (EMG). Previously, pain recognition on the BVDB has been framed as a supervised classification problem, requiring labeled data of different pain levels. While this approach has been successful, it is limited by the availability of annotated data and its generalizability to detect other anomalies beyond pain. This study pivots from traditional supervised classification to unsupervised anomaly detection (AD), which is a more general problem that does not require labeled data of pain anomalies. AD methods are designed to model the normal behavior of the data during training, enabling the detection of anomalies as deviations that fall outside the learned patterns. Here, three AD methods were trained exclusively on normal BVDB time series data, while the pain anomalies provide a realistic test case for evaluating the methods' effectiveness in detecting a subset of the anomaly space. Alternatively, these AD methods can be employed for preprocessing tasks such as data cleaning.

One of the primary difficulties is the high variability of the normal signals across different subjects. Additionally, biophysiological signals often contain noise, artifacts, and non-stationarity. Therefore the challenge lies in selecting an appropriate AD

method that effectively captures the underlying normal patterns while remaining sensitive to deviations indicative of pain. This work evaluates three distinct types of AD methods for pain recognition in biophysiological time series data: Matrix Profile (distance-based), Isolation Forest (density/isolation-based), and Autoencoder (prediction/ reconstruction-based).

Recent studies, such as [6, 7], have criticized current practices in time series anomaly detection. [7] highlighted issues, including using point-adjusted F1 as an evaluation metric, increasing model complexity without significant performance gains, and the use of unrealistic benchmarks. Similarly, [6] noted flaws in commonly used datasets, such as triviality, unrealistic anomaly density, mislabeled ground truth, and biases that distort evaluation outcomes. Both studies emphasized the need for more realistic datasets and simpler, more interpretable methods. In this context, the BVDB is a suitable choice because it offers a realistic and challenging dataset, where even classification models struggle to achieve accuracy levels above 85% for EDA and 60% for ECG and EMG [3].

The evaluation considers two different scenarios: cross-subject evaluation, where models are trained on normal data from a set of subjects and tested on unseen subjects, and subject-specific evaluation, where models are trained using a small number of normal samples from a single subject (approximately 100 seconds) and tested on the same subject.

The key contributions of this work are as follows:

- Evaluation of three AD methods (Matrix Profile, Isolation Forest, and Autoencoder) for identifying pain-related anomalies in the BVDB dataset.
- Comparison of the performance between the unsupervised AD methods themselves and against a supervised classification model to highlight the strengths and limitations of each approach.
- Visualization of anomalous points or segments detected by the AD methods to enhance the interpretability of the results.

The structure of this paper is as follows: Chapter 2 provides background information. Chapter 3 reviews related work in pain recognition and time series AD. Chapter 4 details the methods employed, including Matrix Profile, Isolation Forest, and Autoencoder. Chapter 5 describes the experimental setup, including the BVDB dataset, procedures, and evaluation metrics. Chapter 6 presents the results, followed by a discussion in chapter 7 and conclusions in chapter 8.

2 Background

Anomaly Detection (AD), also referred to as outlier detection, is the task of identifying data instances that significantly deviate from the majority of data instances. AD has crucial applications across domains such as healthcare, finance, industrial systems, and cybersecurity. These anomalies can indicate critical events, such as system failures, fraudulent activities, or medical issues, that require immediate attention. A fundamental assumption in AD is that the region where the normal data lives is concentrated or clustered [8] and can be bounded by some threshold $\tau \geq 0$:

$$\mathcal{X} \setminus \mathcal{A} = \{x \in \mathcal{X} \mid p^{+}(x) > \tau\}, \tag{2.1}$$

where $\mathcal{X} \in \mathbb{R}^{D}$ denotes the data space, $\mathcal{A}$ represents the set of anomalies, and $p^{+}(x)$ is the probability density function of the normal data distribution. In contrast, anomalies $\mathcal{A}$ are assumed to be sparse and dispersed. Therefore, most AD methods focusing on capturing some high-density subset of the normal data and then utilize some threshold τ to detect anomalies using the decision function:

$$\text{decision} = \begin{cases} \text{anomaly} & \text{if} \quad s(x) \geq \tau \\ \text{inlier} & \text{if} \quad s(x) < \tau, \end{cases} \tag{2.2}$$

where $s(x)$ denotes the anomaly score.

Time series AD deals with ordered data, where the temporal dimension is key. This introduces challenges such as seasonality, nonstationarity, and noise, which necessitate specialized techniques [9]. Time series data can be univariate, involving a single signal, or multivariate, encompassing multiple signals with potential dependencies and correlations. The following sections outline types of anomalies in time series data (section 2.1), levels of supervision in AD (section 2.2), and threshold selection methods to determine the boundary between normal and anomalous scores (section 2.3).

2.1 Types of Anomalies in Time Series

Anomalies are often categorized into three broad types: point, contextual, and collective anomalies [10, 11, 9]. The choice of detection method often depends on the type of anomaly, as certain methods are better suited for specific types [12].

- A **point anomaly** (or global point anomaly) refers to a data point or a subsequence that abruptly deviates from the rest of the data points. Examples include missing values or a spike that takes on extreme values, caused by temporal noise, sensor error or events that could lead to serious consequences. For detection, setting a simple upper and lower control limit is typically sufficient.

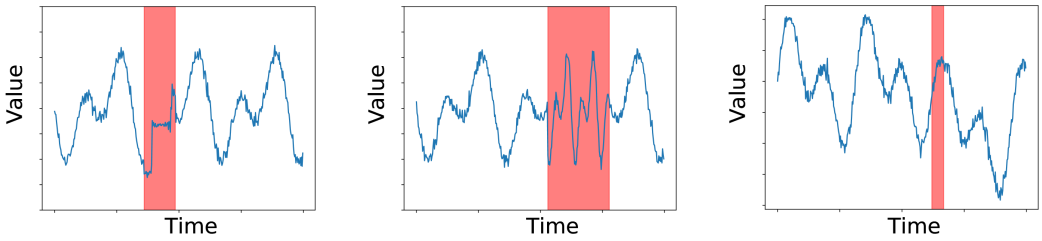
- A **contextual anomaly** (or local point anomaly) refers to a data point or a subsequence that deviate within their local context (neighboring data points within certain ranges) while appearing normal globally. For detection, the deviation could be calculated between the actual and the expected value, while the expected value could be the output of a regression model or simply the mean of the context window.

- A **Collective anomaly** (or pattern anomaly) refers to a subsequence of points that do not repeat a previously observed pattern. The individual data points in a collective anomaly may not be unusual, but their occurrence together as a collection is anomalous. A subsequence $X_{i:j}$ of a time series X can be formulated such that

$$X_{i:j} = \rho(2\pi\omega\tau_{i,j}) + \tau(\tau_{i,j}), \tag{2.3}$$

  where ρ represents shape, ω seasonality, τ trend, and $\tau_{i,j}$ time steps. Collective anomalies are further categorized into shapelet, seasonal, or trend anomalies (Fig. 2.1). Detecting these anomalies typically requires a more advanced pattern recognition method, such as convolutional approaches.

In multivariate time series data, variables/channels often contain dependencies and correlations. Despite this, univariate AD methods are often effective by analyzing each variable independently and then combining the results. This approach works well when anomalies are detectable in individual series and especially when multiple signals show anomalies for the same points or subsequences, enhancing confidence in detection. However, if anomalies are only apparent through the relationships between variables, dedicated multivariate AD methods become necessary.



(a) Shapelet anomaly (b) Seasonal anomaly (c) Trend anomaly

Figure 2.1: Examples of three types of collective/ pattern anomalies (Zhong et al. [13]).

2.2 Supervision

Data labeling is an expensive and time-consuming task, making it infeasible for most AD use cases to collect all possible anomaly instances that represent the full range of anomalies. Therefore, fully supervised AD methods are uncommon. Furthermore, the fully supervised approach would essentially reduce the problem to a binary classification task, rather than being recognized as an AD problem [8]. A typical approach is either unsupervised or semi-supervised.

In the unsupervised setting are n unlabeled data instances $x_1, \ldots, x_n \in \mathcal{X}$ available for training a model. These unlabeled data instances are usually collected during a regular operation of a data generating system and are assumed to be normal. However, the data may contain some noise, such as inherent measurement uncertainties, which can make it challenging to accurately estimate the ground-truth level sets of the normal data. Additionally, the data may be contaminated (or polluted) with undetected anomalies, leading to a distorted decision boundary for the normal data.

In the semi-supervised setting are additionally to the n unlabeled data instances $x_1, \ldots, x_n \in \mathcal{X}$ also m labeled data instances $(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_m, \tilde{y}_m) \in \mathcal{X} \times \mathcal{Y}$ available for training a model. The set of classes $\mathcal{Y}$ consists of the normal and/or the anomalous class. Labeled instances are usually far fewer than unlabeled ones due to the cost of labeling and the scarcity of anomalies in real-world datasets. Recent methods have shown that even a small set of labeled anomalies can significantly improve detection performance [14, 15]. Nontherless, this study focuses only on the unsupervised setting, where no labeled anomalies are available.

2.3 Threshold Selection

AD methods output anomaly scores that, when normalized, are in the range of $[0, 1]$, where a higher score indicates a higher likelihood of being an anomaly. To make a binary decision, a threshold τ is needed to separate normal data from anomalies. Therefore, the threshold selection is another crucial aspect of AD, that influences the accuracy, recall, precision and other threshold dependent metrics. The threshold can be set manually, based on domain knowledge, or automatically, based on the the normal training data. The following provides an overview of common threshold selection methods reviewed in [16], along with additional approaches related to those utilized in this study.

Automatic threshold selection methods can be broadly categorized into supervised and unsupervised approaches. In practice, the AD system may first select a threshold in an unsupervised manner and then adaptively fine-tune it, once labeled anomalies becomes available. Supervised threshold selection methods rely on labeled data where anomalies are explicitly identified and maximize a specific evaluation metric. One common supervised method is using the Receiver Operating Characteristics (ROC) curve, which plots the true positive rate (TPR) against the false positive rate (FPR) for different thresholds. The optimal threshold is selected by maximizing the differences:

$$\tau = \arg\max_{\tau_i} \left(\mathsf{TPR}(\tau_i) - \mathsf{FPR}(\tau_i)\right), \quad i = 1, \ldots, n, \tag{2.4}$$

where n is the number of thresholds considered. Another supervised approach uses the Precision-Recall (PR) curve, which plots the precision against the recall for different thresholds:

$$\tau = \arg\max_{\tau_i} \frac{(1+\beta^2) \cdot \mathsf{Precision}(\tau_i) \cdot \mathsf{Recall}(\tau_i)}{\beta^2 \cdot \mathsf{Precision}(\tau_i) + \mathsf{Recall}(\tau_i)}, \quad i = 1, \ldots, n, \tag{2.5}$$

where β is a parameter that controls the trade-off between precision and recall. For $\beta = 1$, the F1 score is maximized. Similarly, other metrics, such as accuracy, can be maximized using the supervised approach.

In contrast, unsupervised threshold selection methods operate without labeled data, using statistical characteristics of the dataset. The percentile method is a simple parametric approach that sets the threshold as the p-th percentile of the anomaly scores:

$$\tau = \mathsf{percentile}(s(X), p), \tag{2.6}$$

where $s(X)$ is the set of normal anomaly scores and $1-p$ is the expected percentage of anomalies in the data.

There exist also several non-parametric unsupervised threshold selection methods that are used in practice. A popular method is based on the interquartile range (IQR):

$$\tau = Q_3(s(X)) + 1.5 \cdot IQR, \quad IQR = Q_3(s(X)) - Q_1(s(X)) \tag{2.7}$$

The K-Means-based threshold method [17] uses clustering to partition anomaly scores into three clusters: normal, transitional, and anomalous states. The initial threshold is set to the minimum value of the anomalous cluster. Then threshold candidates are generated incrementally to evaluate and refine the ooptimal threshold using the goodness measure:

$$S = \frac{\sigma(s(X))}{\sigma(s(X)_l) \cdot |s(X)_h|}, \tag{2.8}$$

where $\sigma(s(X))$ is the standard deviation of all scores, $\sigma(s(X)_l)$ is the standard deviation of scores below the threshold, and $|s(X)_h|$ is the count of scores above it. The final thresold is the candidate with the highest S. Another method uses normal and non-normal mixture models (MIXMOD) [18] to estimate the threshold. MIXMOD applies a two-component mixture model to the anomaly scores, combining distributions like normal, uniform, pareto and others. Using the Expectation-Maximization (EM) algorithm and maximum likelihood estimation (MLE), the model identifies the threshold at the point where the posterior probabilities of the two components, normal and anomalous, are equal.

In conclusion, threshold selection methods range from simple to sophisticated ones. While some methods rely on explicit supervision or parameters, others infer thresholds in a data-driven way. However, some form of knowledge about the anomalies is necessary, whether provided through supervision, embedded in a parameter, or inherently assumed by the chosen method.

3 Related Work

This chapter presents an overview of previous work on time series classification and anomaly detection (AD) methods, with a focus on physiological signals. The chapter is divided into two sections: time series classification in section 3.1 and time series AD in section 3.2.

3.1 Time Series Classification

In contrast to time series AD methods, time series classification methods aim to model distinct classes by capturing the characteristics of each class in a supervised manner to differentiate between them. However, both methods share the challenge of extracting relevant features from data with temporal dependencies. Typical approaches to feature extraction in time series data include measuring distances between sequences, calculating descriptive statistics that summarize entire sequences or subsequences, applying convolutions and pooling operations, and using neural networks, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs). [19, 20] concluded that the best-performing methods for time series classification in their datasets are dictionary-based ROCKET and HIVE-COTE v2.0. ROCKET [21] (RandOm Convolutional KErnel Transform) uses random convolutional kernels to transform time series data, while HIVE-COTE [22] (Hierarchical Vote Collective of Transformation Ensembles) is an ensemble of classifiers that also includes ROCKET.

For research on automatic pain assessment methods, the BioVid Heat Pain Database (BVDB) [5] is a popular dataset, particularly for pain intensity estimation and pain classification tasks. This work focuses on the physiological signals (EDA, ECG, EMG) of Part A, excluding video data. Most previous experiments have addressed binary classification between the baseline (T_0) and each pain level (T_1 to T_4), though some have performed multiclass classification. This study evaluates AD methods on binary classification between the baseline (T_0) and the highest pain level (T_4). Previous methods

were typically evaluated using Leave-One-Subject-Out (LOSO) cross-validation with all available subjects and samples. A comparison of earlier work on the BVDB for the task T_0 vs. T_4 is shown in Table 3.1. Many experiments show a clear order of importance of signals, with EDA consistently identified as the most informative signal, followed by ECG and then EMG. Multiple studies report that using EDA alone results in the best performance [3, 23]. This led to some studies to focus exclusively on EDA [24, 25], while others incorporate ECG alongside EDA [26, 27].

Many previously published approaches for pain detection on the BVDB time series data train classifiers on hand-crafted features. In [2], researchers collected and analyzed 135 features across categories such as amplitude, frequency, stationarity, entropy, linearity, and variability. A common strategy for pain detection involves training conventional machine learning models, such as Support Vector Machines (SVMs) or Random Forest (RF) classifiers, on these features. [27] compared the performance of various tree-based classifiers, including RF, Adaptive Boosting (AdaBoost), eXtreme Gradient Boosting (XGBoost), and TabNet, identifying TabNet as the best-performing model using EDA and ECG features. In [23], the authors trained other machine learning models, including linear regression, K-Nearest Neighbor (KNN), Support Vector Regression (SVR), and Neural Networks. They performed both standard subject-independent LOSO cross-validation and subject-dependent evaluations, where models are trained and tested on different data samples from the same person. A hybrid model combining these approaches showed that ECG and EMG signals benefited from clustering-based personalization, while EDA was more suitable for generic pain assessment models.

There exists also some hybrid approaches that combine feature engineering with deep learning or feature learning with classical machine learning. In [24], EDA feature extraction methods based on hand-crafted features and learned features with supervised and unsupervised methods were compared. A RF classifier was chosen to compare each feature extraction technique. Key findings indicate that simple feature engineering performs competitively with deep learning methods, and more complex deep learning architectures do not necessarily yield better results than simpler ones. [28] conducted further research in this direction and found that for EDA, hand-crafted features and deep learning features focus on straightforward characteristics: rises in EDA are strongly associated with pain, while stable or declining trends suggest no pain. They demonstrated that the EDA feature "ArgMax" alone provides a strong baseline accuracy of 82.87%, by testing whether the maximum value is located after a threshold of 3.85 seconds in a test sample.

Fully deep learning-based methods are also common for feature extraction and classification. [3] applied a CNN for pain detection, showing that EDA alone yielded the best performance. For multiple signals, a late fusion approach based on weighted averages was proposed. To better capture temporal patterns, subsequent studies incorporated Long Short-Term Memory (LSTM) networks [29] or Bidirectional LSTM (BiLSTM) networks [26] alongside CNNs. [30] proposed a Deep Denoising Convolutional Auto-Encoder (DDCAE) with a gating layer to combine latent representations from multiple signals, with attention mechanisms further enhancing performance. [25] introduced PainAttnNet, comprising multiscale convolutional networks, a squeeze-and-excitation residual network, and a transformer encoder block. Overall, comparisons of different methods for pain detection on the BVDB dataset (Table 3.1) indicate that deep learning methods slightly outperform conventional machine learning techniques.

Table 3.1: Previous work on the BioVid Heat Pain Database for the task T_0 vs. T_4.

Method	**Year**	**Signals**	**Accuracy (%)**
CNN [3]	2019	EDA	84.57 ± 14.13
DDCAE [30]	2021	EDA, ECG, EMG	84.25 ± 13.82
SVM [23]	2021	EDA	83.3
RF [28]	2023	EDA	83.56 ± 14.94
CNN + BiLSTM [26]	2023	EDA, ECG	84.8 ± 13.3
PainAttnNet [25]	2023	EDA	85.34

3.2 Time Series Anomaly Detection

Numerous methods for time series AD have been developed across various research domains. This section provides an overview of popular time series AD methods in general and their application to physiological signals. Similar to [31], the methods are categorized into three main groups: distance-based in subsection 3.2.1, density-based in subsection 3.2.2, and prediction-based in subsection 3.2.3.

3.2.1 Distance-based

Distance-based methods typically utilize an explicit distance measure to compare sequences or subsequences with a reference pattern. Anomalies are then detected based on their greater distance from their nearest neighbor. These methods are generally unsupervised and require no training. Dynamic Time Warping (DTW) is a widely used distance-based algorithm that calculates the optimal alignment between two sequences by minimizing cumulative distance. DTW is applied in [32] for AD in ECG signals, where a human expert provides normal patterns to enhance the confidence in the decision process. Clustering methods such as k-Means [33] or SAND [34] are also part of the distance-based category. For AD with k-Means, subsequences are first clustered, and distances between subsequences and their cluster centroids are measured. SAND improves on this approach by aligning sequences before calculating similarity. Another distance-based method involves matrix profile-based approaches like STAMP (Scalable Time series Anytime Matrix Profile) [35], which efficiently computes the Euclidean distance of each subsequence to its nearest neighbor. DAMP [36], a highly optimized variant, supports online AD on data streams exceeding 300,000 Hz on standard hardware.

3.2.2 Density-based

Rather than explicitly measuring nearest neighbor distances, density-based methods estimate the density of normal data and detect anomalies as regions with low density. In [37], an autoencoder is used to train a CNN encoder that extracts relevant features from ECG signals. These features are then fed into a multivariate Gaussian distribution model to estimate the mean vector and covariance matrix of the normal data. The anomaly score is calculated as the likelihood of a data point under the estimated distribution. Another study [38] evaluated various AD methods for heart rate data, including Local Outlier Factor (LOF), Isolation Forest (IF), and other supervised methods. LOF [39] computes the local density of data points and compares them with the densities of neighboring points. IF [40] is an ensemble method that isolates anomalies by iteratively partitioning data until each point is isolated. Anomalies generally require fewer splits and are thus detectable. [41] compared IF and RF, finding that RF outperformed IF, while [40] demonstrated better performance for IF over RF. These differences stem from dataset characteristics and anomaly types. When anomalies are clustered or adequately represented by limited training samples, supervised classification approaches generally perform better. Extensions of IF, such

as Deep Isolation Forest (DIF) [42], use casually initialized neural networks to generate random representations of data, enabling non-linear partitioning with axis-parallel cuts. DIF demonstrated improved performance, including for vectorized ECG time series generated via TS2Vec [43] contrastive learning.

3.2.3 Prediction-based

Prediction-based methods involve training a model to predict normal temporal behavior and using prediction error as the anomaly score. These methods can be divided into two subcategories: reconstruction-based and forecasting-based methods.

Reconstruction-based methods aim to train a model capable of reconstructing normal data from a compressed or corrupted state, under the assumption that anomalies will exhibit higher reconstruction errors. Principal Component Analysis (PCA) is a popular dimensionality reduction technique that can be inverted for AD to reconstruct original data with some loss [44]. [45] used a convolutional autoencoder for stress and affect detection on ECG and EDA signals, setting the anomaly score as the multiplied reconstruction error of the separately processed signals. They observed that anomalous behavior was highly subject-dependent. [46] combined CNN and LSTM architectures with a two-stage sliding window to improve feature extraction in autoencoders. BeatGAN [47] used a Generative Adversarial Network for reconstruction and was evaluated on ECG time series. Recently, diffusion models have been applied to time series AD, training models to denoise corrupted time series and smooth abnormal segments [48]. ImDiffusion [49] reduces uncertainty by leveraging neighboring values to condition the denoising process, using grating data masking to employ diffusion models for imputing missing values. [50] proposed Time-VQVAE-AD, which calculates anomaly scores in the latent space rather than input/output space. The method follows the two-stage training approach: first, the input is converted into time-frequency domain and a Vector Quantized Variational Autoencoder (VQ-VAE) is trained by minimizing the reconstruction loss. Second, the prior of the discrete latent space is learned via masked modeling with a bidirectional transformer. TimeVQVAE-AD showed improved performance in detecting small amplitude anomalies by not relying on the error between the target time series and reconstructed time series.

Forecasting-based methods predict future values based on previous observations, using forecasting errors as anomaly scores. Statistical approaches like AutoRegressive Integrated Moving Average (ARIMA) models are commonly applied for time series forecasting [51]. However, recent research predominantly focuses on deep learn-

ing models such as Recurrent Neural Networks (RNNs) or Transformer. LSTM-AD [52] proposed a stacked two-layer LSTM network for AD in periodic signals like ECG readings. [53] proposed a transformer-based architecture for ECG AD, incorporating an embedding layer, two bidirectional transformer encoder layers, and final dense layers.

4 Methods

The following sections introduce the AD methods used in this study. section 4.1 describes the Matrix Profile method as a distance-based approach. section 4.2 introduces the Isolation Forest method as an isolation- or density-based approach. section 4.3 describes the Autoencoder method as a prediction- or reconstruction-based approach.

4.1 Matrix Profile

The Matrix Profile method (MP) for time series AD is a distance-based approach that measures shape-based similarity between two time series: a test sample T_A to be evaluated and a reference sample T_B, which consists of the concatenation of k previously known normal samples. The two time series are considered similar if they share many similar subsequences under the z-normalized Euclidean distance. Z-normalization removes differences in mean μ and standard deviation σ between subsequences, focusing solely on their shape.

As its name suggests, the MP method is based on the computation of the matrix profile P_{AB}. The Matrix Profile P is widely used in data mining for time series analysis, particularly due to the development of efficient and scalable algorithms like STAMP (Scalable Time series Anytime Matrix Profile) [35]. P is a meta time series encoding information useful for tasks such as motif discovery, discord discovery, shapelet discovery, and semantic segmentation. For a single time series T (self-join), the matrix profile P is computed by iterating over every subsequence of length m and finding its (non-trivial-match) nearest neighbor subsequence in T. Anomalous subsequences (discords) in T are identified by having higher distances to their nearest neighbors compared to other subsequences. The self-join can be generalized to an AB-join, which records the nearest neighbor of subsequences in T_A to subsequences in T_B. Here, T_A of length n is the test time series, and T_B, of any length, defines the normal behavior. Specifically, T_B is the concatenation of the k most normal samples, where

the optimal k is determined experimentally and varies by signal. The resulting P_{AB} of length $n - m + 1$ is converted to a single value to obtain the anomaly score for T_A. The 70^{th}-percentile of P_{AB} was found to be an effective measure. An illustration of the MP method is provided in Figure 4.1.

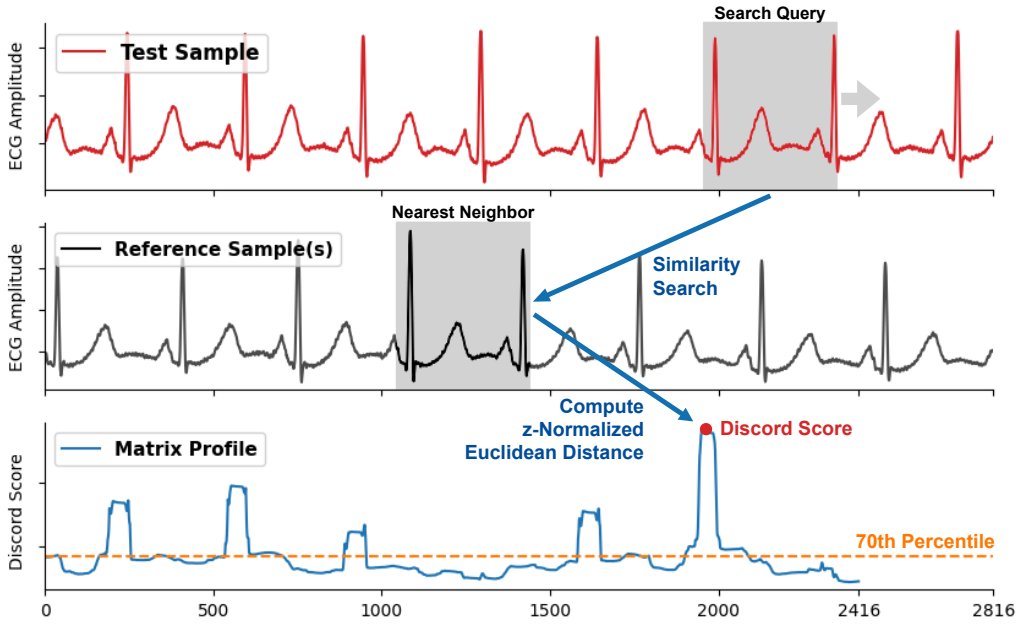


Figure 4.1: Illustration of the MP method for the ECG signal with window size $m = 400$: For each window of the test sample, the distance to the nearest neighbor window in the k normal/reference samples is calculated (here $k = 1$). The reported anomaly score is the 70^{th} percentile of the matrix profile.

A straightforward but computationally expensive approach to compute P_{AB} involves calculating the entire distance matrix between all subsequences, with a time complexity of $O(n^2 m)$ and a space complexity of $O(n^2)$:

1. Extract all subsequences $T_{A_{i,m}}$ and $T_{B_{j,m}}$ of length m starting from position i.

2. Compute all pairwise z-normalized Euclidean distances:

$$d_{i,j} = \sqrt{\sum_{l=0}^{m-1} \left(\frac{T_{A_{i+l}} - \mu_i}{\sigma_i} - \frac{T_{B_{j+l}} - \mu_j}{\sigma_j} \right)^2}, \tag{4.1}$$

where μ_i and σ_i are the mean and standard deviation of $T_{A_{i,m}}$, and μ_j and σ_j are the mean and standard deviation of $T_{B_{j,m}}$. Store these distances in a distance matrix $M_{i,j}$.

3. Extract the minimum from each row (distance profile) in the distance matrix and store it in a vector P_{AB}:

$$P_{AB,i} = \min_j M_{i,j} \quad \text{for } 1 \leq i \leq n - m + 1. \tag{4.2}$$

To compute P more efficiently, the STOMP (Scalable Time series Ordered-search Matrix Profile) algorithm [54] is used. STOMP leverages the Fast Fourier Transform (FFT) and ordered search to achieve a time complexity of $O(n^2)$ and a space complexity of $O(n)$. The procedure of STOMP, as proposed in [54], is outlined in algorithm 1.

Algorithm 1 STOMP (T, m)

Require: A time series T and a subsequence length m
Ensure: Matrix profile P and the associated matrix profile index I of T

1: $n \leftarrow$ Length(T), $l \leftarrow n - m + 1$
2: $\mu, \sigma \leftarrow$ ComputeMeanStd(T, m)
3: $QT \leftarrow$ SlidingDotProduct$(T[1:m], T)$, $QT_{\text{first}} \leftarrow QT$ ▷ using FFT
4: $D \leftarrow$ CalculateDistanceProfile(QT, μ, σ) ▷ using Eq. 4.3
5: $P \leftarrow D$, $I \leftarrow$ ones ▷ initialization
6: **for** $i = 2$ to l **do** ▷ in-order evaluation
7: **for** $j = l$ downto 2 **do** ▷ update dot product using Eq. 4.4
8: $QT[j] \leftarrow QT[j-1] - T[j-1] \times T[i-1] + T[j+m-1] \times T[i+m-1]$
9: **end for**
10: $QT[1] \leftarrow QT_{\text{first}}[i]$
11: $D \leftarrow$ CalculateDistanceProfile(QT, μ, σ, i) ▷ using Eq. 4.3
12: $P, I \leftarrow$ ElementWiseMin(P, I, D, i)
13: **end for**
14: **return** P, I

The algorithm for STOMP begins in line 1 by computing the length l of P. Line 2 precalculates μ and σ for every subsequence. Line 3 calculates the first dot product vector QT using the sliding dot product, which can be viewed as a convolution operation. To compute the convolution in $O(n \log n)$ time, the Fast Fourier Transform (FFT) is used. FFT leverages the convolution theorem, which states that the convolution of two sequences in the time domain is equivalent to the point-wise multiplication of their Fourier transforms in the frequency domain [55]. Line 5 calculates the z-normalized Euclidean distance profile D of the first ($i = 1$) subsequence $T_{i,m}$ to all other sub-

sequences $T_{j,m}$ using their dot products $QT_{i,j}$ and the precalculated μ and σ. The equation is given by:

$$d_{i,j} = \sqrt{2m\left(1 - \frac{QT_{i,j} - m\mu_i\mu_j}{m\sigma_i\sigma_j}\right)}. \tag{4.3}$$

Line 6 initializes P and the matrix profile index I. Lines 7-12 calculate the distance profile D of every subsequence of T in sequential order and store the minimum distance in P and the corresponding index in I. However, the dot product QT does not need to be recalculated using FFT. Due to locality, it can be updated as follows:

$$QT_{i,j} = QT_{i-1,j-1} - t_{i-1}t_{j-1} + t_{i+m-1}t_{j+m-1}, \tag{4.4}$$

where t_i is the i^{th} element of T. Therefore, all n distance profiles can be calculated in $O(n)$ time, resulting in a total time complexity of $O(n^2)$. Finally, line 13 returns the matrix profile P and the matrix profile index I of T. The matrix profile index I stores the location of the nearest neighbor for each subsequence in the reference time series. However, I is not utilized in this study.

4.2 Isolation Forest

The Isolation Forest (IF) method [40] for AD is a density- or isolation-based approach designed to isolate anomalies directly, rather than modeling the normal data distribution. Unlike the MP method, which processes raw time series data, IF requires tabular data. This tabular data is constructed by extracting hand-crafted features from the time series data. Summarized in Table 4.1, 44 features were extracted (23 for EDA, 12 for ECG, and 9 for EMG). These features were gathered from prior studies on the BVDB dataset and their feature rankings or importance. Following [2], the features are categorized into statistical groups (amplitude, variability, frequency, entropy, and higher-order statistics) to indicate the types of information they capture. To optimize the feature set for IF training, feature importance was assessed using impurity scores derived from a Random Forest classifier. The top n features were incrementally evaluated, starting with the most important feature and progressively adding features to identify the subset that yielded the best IF performance.

IF operates by constructing isolation trees (iTrees) that recursively partition the feature space to isolate samples. Since recursive partitioning can be represented by

a tree structure, the number of partitions required to isolate a sample corresponds to the path length from the root node to the terminating node. Anomalies tend to have shorter path lengths, as illustrated in Figure 4.2. IF has a time complexity of $O(t\psi \log \psi)$ for training and $O(nt \log \psi)$ for evaluation, where t is the number of trees, n is the number of instances, and ψ is the subsampling size. Subsampling reduces computational cost, enhances tree diversity, and mitigates masking (where dense clusters of anomalies hide individual anomalies) and swamping (where normal instances are wrongly identified as anomalies).

During training, iTrees are constructed as described in 2. Lines 5-6 randomly select a feature q and a split point p. Lines 7-8 partition the subsample X into subsets X_l and X_r based on the split point. This process is recursively repeated (lines 9-12) until the tree reaches a height limit l or a single instance is isolated. To construct a forest, multiple iTrees are built using different subsamples of the training data. During evaluation, instances traverse each iTree, and the number of visited edges e is recorded to calculate the path length $h(x)$. The anomaly score $s(x, n)$ is determined based on the average path length $E(h(x))$ across all trees:

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}}, \tag{4.5}$$

where $c(n)$ is the average path length of unsuccessful searches in a binary search tree:

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n}, \tag{4.6}$$

with $H(i)$ as the harmonic number, approximated by $ln(i) + 0.5772156649$ (Euler's constant).

Table 4.1: Features extracted from the physiological signals.

Name	Signals	Group	Feature Description	Ref
Max	EDA, EMG	Amplitude	Maximum value.	[57]
Min	EDA	Amplitude	Minimum value.	[57]
ArgMax	EDA	Amplitude	Index of the maximum value.	[28]
ArgMin	EDA	Amplitude	Index of the minimum value.	[28]
Mean	EDA	Amplitude	Average value.	[58]
NormMean	EDA	Amplitude	Normalized mean.	[28]
StD	EDA	Variability	Standard deviation.	[2]

Name	Signals	Group	Feature Description	Ref
NormStD	EDA	Variability	Normalized standard deviation.	[28]
Var	EDA, EMG	Variability	Variance.	[2]
NormVar	EDA	Variability	Normalized Variance.	[28]
Range	EDA	Variability	Difference between the maximum and minimum values.	[2]
DiffStartEnd	EDA	Variability	Difference between the first and last values.	[28]
IqR	EDA, EMG	Variability	Interquartile range.	[57]
MeanPhasic	EDA	Amplitude	Average of the phasic (rapid) component.	[28]
StDPhasic	EDA	Variability	Standard deviation of the phasic component.	[28]
MinPhasic	EDA	Amplitude	Minimum value of the phasic component.	-
MeanTonic	EDA	Amplitude	Average of the tonic (slow) component.	[28]
StDTonic	EDA	Variability	Standard deviation of the tonic component.	[28]
RangeTonic	EDA	Amplitude	Range (max - min) of the tonic component.	[28]
Skew	EDA	Higher order statistics	Skewness (Asymmetry Of The Distribution).	[58]
Kurtosis	EDA	Higher order statistics	Kurtosis (Tailedness Of The Distribution).	[58]
Max1D	EDA	Variability	Maximum value of the first derivative.	[27]
Min1D	EDA	Variability	Minimum value of the first derivative.	[27]
NumRPeaks	ECG	Frequency	Count Of R-Peaks in the Signal.	[57]
MIBIs	ECG	Variability	Average of the inter-beat intervals (R-R intervals).	[27]
SlopeRR	ECG	Variability	Average slope of the linear regression line of the R-R intervals.	[2]
MeanRR	ECG	Amplitude	Average of the R-R intervals.	[2]
StDRR	ECG	Variability	Standard deviation of the R-R intervals.	[57]
RMSSD	ECG	Variability	Root mean square of successive differences of R-R intervals.	[2]
RyStD	ECG	Variability	Standard deviation of the R amplitudes.	[27]
RyRange	ECG	Amplitude	Range (max - min) of R amplitudes.	[27]
PTxStD	ECG	Variability	Standard deviation of P-T durations.	[27]
PTxRMSSD	ECG	Variability	Root mean square of successive differences of P-T durations.	[27]
PTyStD	ECG	Variability	Standard deviation of P-T amplitudes.	[27]
PTyRMSSD	ECG	Variability	Root mean square of successive differences of P-T amplitudes.	[27]

Name	Signals	Group	Feature Description	Ref
ZC	EMG	Frequency	Count of zero crossings	[2]
MAV	EMG	Amplitude	Mean absolute value.	[58]
RMS	EMG	Amplitude	Root mean square.	[58]
StDStD	EMG	Variability	Standard deviation of a vector of standard deviations from segmented signals.	[2]
ApEn	EMG	Entropy	Approximate entropy, measuring signal regularity.	[2]
ShannonEn	EMG	Entropy	Shannon entropy, measuring signal complexity.	[2]

4.3 Autoencoder

An Autoencoder (AE) is a neural network architecture commonly used for unsupervised learning [59]. It consists of two main components: an encoder network ϕ_e that compresses the input data x into a lower-dimensional representation $z = \phi_e(x; \mathcal{W}_e)$, and a decoder ϕ_d that reconstructs the input data $\hat{x} = \phi_d(z; \mathcal{W}_d) \approx x$. The objective [60] is to learn weights $\mathcal{W}_e^*$ and $\mathcal{W}_d^*$ that minimize the reconstruction error:

$$\{\mathcal{W}_e^*, \mathcal{W}_d^*\} = \underset{\mathcal{W}_e, \mathcal{W}_d}{\arg\min}\, E[\Delta(x, \phi_d(\phi_e(x; \mathcal{W}_e); \mathcal{W}_d))], \tag{4.7}$$

where E is the expectation over the distribution of x and Δ is a loss function that measures the reconstruction error as the difference between the input x and the output $\hat{x}$. Here, the mean squared error (MSE) loss function is used after the last layer of the decoder, which is for defined as:

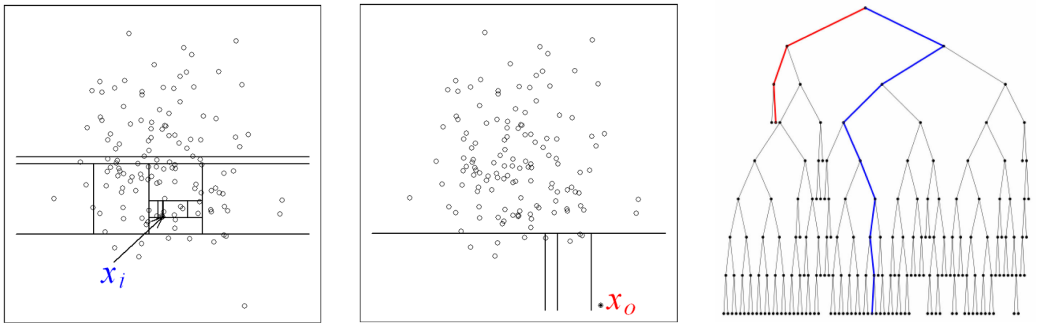
$$\Delta = \text{MSE} = \frac{1}{TC} \sum_{t=1}^{T} \sum_{c=1}^{C} (x_{t,c} - \hat{x}_{t,c})^2, \tag{4.8}$$

where T is the number of time steps, C is the number of channels, and $x_{t,c}$ and $\hat{x}_{t,c}$ are the input and output values at time step t and channel c.

In the context of AD, the AE is trained on normal data only. Consequently, when presented with anomalous data during inference, the reconstruction error is expected to be higher due to deviations from the learned normal patterns. The anomaly score is therefore set as the reconstruction error.

Algorithm 2 $iTree(X, e, l)$

Require: X - input data, e - current tree height, l - height limit
Ensure: an $iTree$
if $e \geq l$ **or** $|X| \leq 1$ **then**
  return $exNode\{Size \leftarrow |X|\}$
else
  Let Q be a list of features in X
  Randomly select a feature $q \in Q$
  Randomly select a split point p from max and min values of feature q in X
  $X_l \leftarrow filter(X, q < p)$
  $X_r \leftarrow filter(X, q \geq p)$
  return $inNode\{Left \leftarrow iTree(X_l, e+1, l),$
    $Right \leftarrow iTree(X_r, e+1, l),$
    $SplitAtt \leftarrow q,$
    $SplitValue \leftarrow p\}$
end if



(a) Isolating x_i (b) Isolating x_o (c) Isolation Tree

Figure 4.2: Visualization of the branching process for a normal data point x_i (blue) and an anomalous data point x_o (red). a) and b) show the recursive partitioning by randomly selecting a feature and then randomly selecting a split value until the point is isolated. c) represents the branching process in a binary tree. Anomalies tend to have shorter path lengths or require fewer splits. ((a, b) Liu et al. [40], (c) Hariri et al. [56]).

In the experiments, a convolutional AE is used, where the encoder consists of 1D convolutional layers (Conv1d), and the decoder mirrors the encoder with 1D transposed convolutional layers (ConvTranspose1d). The 1D convolutional networks are designed to extract and learn temporal dependencies in the time series by capturing

local patterns. A convolutional layer can be expressed by the following equation:

$$x_j^l = \sum_{i \in M_j} x_i^{l-1} \cdot w_{ij}^l + b_j^l, \tag{4.9}$$

where x_j^l represents the output feature map value corresponding to the $j\text{-}th$ convolution kernel at layer l, x_i^{l-1} is the input feature map value from the previous layer, M_j is the receptive field of the kernel, w_{ij}^l is the learned kernel weight, and b_j^l is the bias term [61]. The following hyperparameters define the main behavior of a convolutional layer:

- **Kernel size**: The length of the filter that slides over the input data.
- **Stride**: For Conv1d, stride controls the downsampling factor by setting the step size for the kernel's movement over the input. A larger stride skips positions, reducing the output size. For ConvTranspose1d, stride controls the upsampling factor by inserting zeros between the input values before applying the convolution.
- **Padding**: The number of values added to the boundaries of the input data to control the output size.
- **No. units/ output channels**: The number of filters or units applied to the input data. Each filter processes all input channels simultaneously, learning one kernel per input channel. For each filter, the outputs of the per-channel convolutions are summed to create a single output channel, allowing the network to learn different aspects of the input data per filter.

For each signal, the AE consists of a slightly different number of layers and dimensions. The architecture of the AE for the ECG signal is shown in Table 4.2. Additional details about the hyperparameters and training process are provided in subsection 5.2.3. A ReLU (Rectified Linear Unit) activation function is applied after each convolutional layer, except for the decoder's last layer, to introduce non-linearity. The ReLU function is defined as:

$$ReLU(x) = \max(0, x). \tag{4.10}$$

In the last layer of the encoder, a linear layer acts as a bottleneck between the encoder and decoder. For ECG and EMG signals, the feature vector at each time point is reduced to the latent dimension size. For EDA signals, the input feature maps are first flattened and then fully connected to the latent dimension size.

Table 4.2: AE architecture (for the ECG signal with a length of 1152).

Encoder			**Decoder**		
Layer	**No. Units**	**Output Size**	**Layer**	**No. Units**	**Output Size**
Conv1d	16	576×16	Linear	-	18×128
Conv1d	32	288×32	ConvTranspose1d	128	36×128
Conv1d	32	144×32	ConvTranspose1d	64	72×64
Conv1d	64	72×64	ConvTranspose1d	32	144×32
Conv1d	128	36×128	ConvTranspose1d	32	288×32
Conv1d	128	18×128	ConvTranspose1d	16	576×16
Linear	-	$18 \times$ latent_dim	ConvTranspose1d	1	1152×1

5 Experimental Apparatus

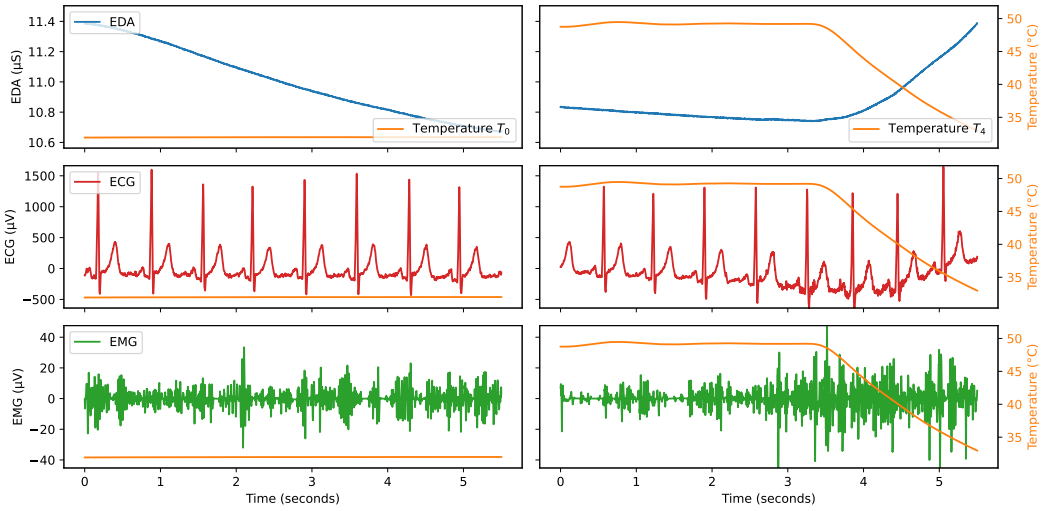
This chapter first presents the dataset used in the study. Subsequently, the procedure of the experiments is described, followed by the metrics used to evaluate the performance of the AD methods.

5.1 BioVid Heat Pain Database

The experiments were conducted with the BioVid (Biopotential and Video) Heat Pain Database[1] (BVDB), which was collected through a collaboration between the Neuro-Information Technology group at the University of Magdeburg and the Medical Psychology group at the University of Ulm [5]. BVDB is divided into different parts, each containing multimodal data collected for pain assessment research. This work utilizes recordings of physiological signals from Part A, excluding frontal video data. The dataset consists of recordings from 87 subjects who underwent heat pain stimulation via a thermode applied to the right forearm. The temperature was adjusted to four different pain levels T_1, T_2, T_3, T_4, in addition to a baseline (no pain) condition at $T_0 = 32\ ^\circ\mathrm{C}$. Pain thresholds were determined individually, where T_1 is the temperature where the subject felt a change from heat to pain, T_4 is the temperature where the pain became unacceptable, and T_2 and T_3 are intermediate levels.

The three recorded physiological signals are as follows: Electrodermal Activity (EDA), also known as Galvanic Skin Response (GSR) or Skin Conductance Level (SCL), which measures skin moisture through electrical conductance; Electrocardiogram (ECG), which measures the electrical activity of the heart; and Electromyogram (EMG), which measures the electrical activity of the trapezius muscles located in the upper back and neck region. These signals are indicative of internal tension, emotional activity, or stress levels, making them relevant for detecting pain-related changes. Each signal was recorded at a sampling rate of 512 Hz over a time window of 5.5 seconds,

[1] `https://www.nit.ovgu.de/BioVid.html`



(a) Baseline (`071309_w_21-BL1-084`) (b) Pain level 4 (`071309_w_21-PA4-068`)

Figure 5.1: Examples of two data samples from the BioVid Heat Pain Database, illustrating the recorded physiological signals (EDA, ECG, and EMG): (a) shows the signals of a normal sample at baseline temperature T_0, while (b) shows the signals of an anomalous sample at the subject-specific pain tolerance temperature T_4.

resulting in 2816 data points per signal. For each subject, the five temperature levels were stimulated 20 times each, resulting in 100 samples (or 300 time series) per subject. For 87 subjects, this totaled 8700 samples. However, in the experiments, only samples from the baseline and pain level 4 were used, with baseline samples treated as normal and pain level 4 samples as anomalous, resulting in 3480 samples (1740 per class).

5.2 Procedure

The general procedure of the experiments is illustrated in Figure 5.2. The input consists of physiological signals from the BioVid Heat Pain Database, which are optionally preprocessed. Features are then extracted from the raw signals, producing a matrix profile, hand-crafted features, or learned features. These features are passed to the AD method to calculate anomaly scores. The autoencoder performs feature extraction and scoring in an end-to-end manner. Finally, a threshold is applied to

Figure 5.2: The pipeline of the experimental procedure.

the anomaly scores to classify the data as normal or anomalous. For thresholding, both fixed percentiles and several parameter-free methods from PyThresh[2] were considered, including Normal and Non-Normal Mixture Models (MIXMOD), Filtering-based (FILTER), Karcher mean/ Riemannian Center of Mass (KARCH), and Elliptical Boundary (EB) thresholder. However, only results using percentiles and the KARCH are reported, as these methods showed the best performance. The anomaly scores and classification results are evaluated using metrics described in section 5.3.

5.2.1 Matrix Profile

The matrix profile was calculated using the STUMPY library. Specifically, the `stump`[3] function was utilized, which computes the matrix profile based on the STOMP algorithm. The time series data was not preprocessed. Evaluation was performed in a subject-specific manner, where the matrix profile was calculated using $k \leq 19$ normal samples and one test sample from the same subject. The window size m was set to 200 for EDA, 400 for ECG, and 500 for EMG. The anomaly score was defined as the 70th percentile of the matrix profile or combined matrix profile. When multiple signals were used, the combined matrix profile was calculated by summing the z-normalized matrix profiles of the individual signals.

5.2.2 Isolation Forest

The physiological time series data was preprocessed before applying the Isolation Forest (IF) method. The EDA signals were smoothed using a Gaussian filter. Following [3], the ECG signals were filtered using a third-order Butterworth bandpass filter with a frequency range of [0.1, 250] Hz. ECG signals were also detrended by subtracting a fifth-degree polynomial least-squares fit. EMG signals were processed

[2] `https://github.com/KulikDM/pythresh`
[3] `https://stumpy.readthedocs.io/en/latest/api.html#stumpy.stump`

Table 5.1: Parameters for Isolation Forest (left) and Random Forest (right).

IF

Parameters	Cross-Subject	Subject-Specific
n_estimators	50	120
max_samples	1%	50%
max_features	100%	70%

RF

Parameters	Cross-Subject
n_estimators	120
max_depth	5
min_samples_split	2
min_samples_leaf	2

with a fourth-order Butterworth bandpass filter with a frequency range of [20, 250] Hz. All signals were subsequently z-normalized using the mean and standard deviation of the normal samples.

The NeuroKit2 library[4] was employed to extract features from the preprocessed signals, such as the location of R-peaks in ECG signals or decomposition of EDA signals into tonic and phasic components. The IF implementation from the scikit-learn library[5] was used to perform AD on the extracted features. A Random Forest (RF) classifier was also trained for comparison and feature ranking based on Gini impurity-based feature importance. Parameters for the IF and RF models were determined through grid search and are provided in Table 5.1.

Experiments with IF included both cross-subject and subject-specific modeling. In cross-subject modeling, the IF model was evaluated using Leave-One-Subject-Out (LOSO) cross-validation. The model was trained on the normal samples of all subjects except one, and tested on the samples of the left-out subject. As a baseline, direct thresholding was performed on the most important feature (`eda_ArgMax`). For subject-specific modeling, the IF model was trained and evaluated on data from the same subject. Each subject's data was evaluated using a 4-fold cross-validation: 15 normal training samples, 5 normal test samples, and 5 anomalous test samples per fold.

5.2.3 Autoencoder

The preprocessing steps for the autoencoder (AE) method were similar to those for Isolation Forest, with additional modifications. For EMG signals, the envelope was

[4] `https://neuropsychology.github.io/NeuroKit`

[5] `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html`

Table 5.2: Hyperparameters for reconstruction methods.

Hyperparameter	PolyFit (EDA)	AE (EDA)	AE (ECG)	AE (EMG)	AE (Multi)
Sampling rate (Hz)	512	32	256	128	256
Trimming	1,000	88	256	0	256
Input time steps	1,816	88	1,152	704	1,152
(kernel size, padding)	-	(3, 1)	(11, 5)	(11, 5)	(5, 2)
Latent size	1	1	18×64	11×2	36×8
Batch size	64	128	128	128	128
Learning rate	0.01	0.001	0.001	0.001	0.001
Epochs	9	100	150	16	80
Trainable parameters	3	5,026	637,121	246,211	157,259

extracted by first rectifying the signal (taking the absolute values) and then smoothing it using a Gaussian filter. To reduce potential mislabeled samples in the training set, the normal samples were reduced by 25% for AE and 50% for PolyFit, using anomaly scores from the Matrix Profile (MP) method to filter outliers. Gaussian noise with a standard deviation randomly selected between 0 and 0.1 and a mean of 0 was added to the ECG signal. To improve performance and reduce computational cost, signals were downsampled, and their initial segments were trimmed.

Hyperparameters for the reconstruction methods are listed in Table 5.2. The autoencoders were implemented using `Conv1d` and `ConvTranspose1d` layers from the PyTorch library[6]. The Adam optimizer [62] was used for efficient stochastic gradient descent. Autoencoders were evaluated in a LOSO cross-validation setting.

For a simple baseline, a polynomial fit (PolyFit) was performed. PolyFit uses a second-degree polynomial parameterized by three coefficients to predict the EDA signal using only the first value of the input signal.

5.3 Metrics

To evaluate the performance of the AD methods, both threshold-independent and threshold-dependent metrics are reported at different stages of the experiments.

The area under the receiver operating characteristic curve (ROC AUC or simply AUC) is used as a threshold-independent metric and is widely regarded as the standard in

[6] `https://pytorch.org/docs/stable/nn.html#convolution-layers`

AD research [63]. AUC represents the probability that a randomly chosen anomaly receives a higher anomaly score than a randomly chosen normal instance [64], thereby measuring the effectiveness of the scoring phase in separating normal and anomalous data. The ROC curve is a plot of the true positive rate (detection probability) against the false positive rate (false alarm probability) for all possible thresholds. AUC is calculated as:

$$\text{AUC} = \int_0^1 \frac{TP}{TP+FN} d\frac{FP}{TN+FP}, \tag{5.1}$$

where true positiv (TP) is the number of correctly classified anomalous data points, true negative (TN) is the number of correctly classified normal data points, false positive (FP) is the number of wrongly classified anomalous data points and false negative (FN) is the number of wrongly classified normal data points. AUC ranges from 0 to 1, where 0.5 implies random guessing, while an AUC of 1 indicates perfect performance.

For pain detection, once a threshold is selected, standard threshold-dependent metrics are reported. These metrics quantify detection performance in practical scenarios and facilitate comparison with previous supervised approaches for pain detection. Accuracy (Acc) measures the proportion of correctly classified instances among all instances:

$$\text{Acc} = \frac{TP+TN}{TP+TN+FP+FN}. \tag{5.2}$$

Additionally, recall (Rec) and precision (Prec) are reported to show the trade-off between capturing all pain instances (high recall) and minimizing false alarms (high precision):

$$\text{Rec} = \frac{TP}{TP+FN}, \quad \text{Prec} = \frac{TP}{TP+FP}. \tag{5.3}$$

6 Results

This chapter presents the results of the AD experiments on the BVDB dataset, organized by method: Matrix Profile in Figure 6.1, Isolation Forest in section 6.2, and Autoencoder in section 6.3.

6.1 Matrix Profile

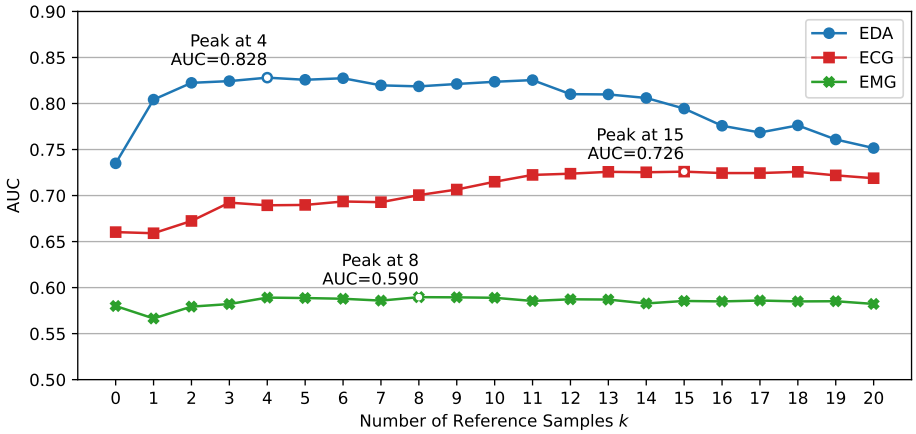
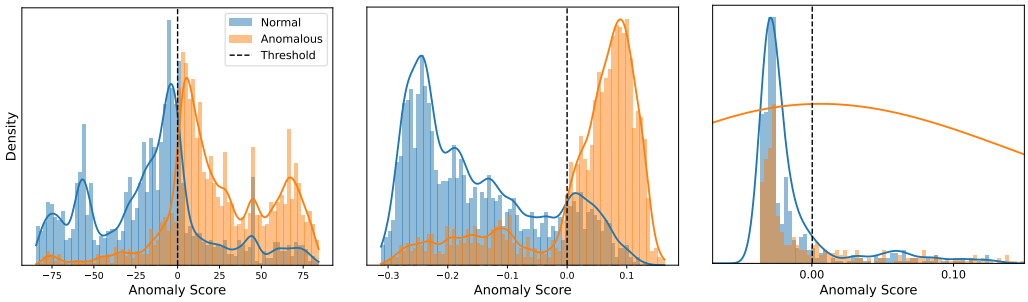


Figure 6.1: AUC performance across different numbers of normal samples k used as reference, for each signal.

For individual signals, the EDA signal showed the best performance with KARCH as the threshold and the number of reference samples $k = 4$, achieving an accuracy of 77.21% and an AUC of 82.33%. ECG performed worse with an accuracy of 66.86% and an AUC of 72.62% using percentile thresholding and setting $k = 15$, while EMG had the lowest performance with an accuracy of only 56.06% and an AUC of 58.23% using KARCH thresholding and setting $k = 8$. Combining signals improved AUC but not accuracy. Ignoring EMG was beneficial, as EDA and ECG combined achieved the best AUC of 83.73%. The results are listed in Table 6.1.

Table 6.1: Results of the MP method for subject-specific anomaly detection: Mean and standard deviation (in %) for each metric across all subjects.

Signals	Thresholding	Acc	Prec	Rec	AUC
EDA	74th-Percentile	73.79±14.65	67.40±20.15	74.42±32.14	82.33±15.67
	KARCH	**77.21±14.88**	75.30±14.42	77.01±26.32	
ECG	69th-Percentile	66.86±13.50	64.17±12.18	65.68±26.98	72.62±18.50
	KARCH	65.17±15.71	66.47±22.60	45.05±31.39	
EMG	58th-Percentile	55.94±11.25	52.95±13.49	54.13±22.87	58.23±15.30
	KARCH	56.06±11.55	54.54±18.87	37.64±23.53	
EDA, ECG, EMG	76th-Percentile	74.65±12.92	71.56±11.58	75.68±25.72	82.46±15.34
	KARCH	75.28±14.83	75.38±13.60	69.54±28.78	
EDA, ECG	76th-Percentile	75.91±12.41	72.65±10.09	**78.04±24.79**	**83.73±14.75**
	KARCH	76.58±14.12	**76.63±12.37**	72.81±26.06	



(a) Matrix Profile (b) Isolation Forest (c) Autoencoder

Figure 6.2: Distribution of the anomaly scores for normal (blue) and anomalous (orange) data across the three methods for EDA. The dashed line represents the supervised accuracy threshold (shifted to zero).

6.2 Isolation Forest

For cross-subject AD, the results are shown in Table 6.2, and the feature ranking by importance (determined using RF) is presented in Figure 6.3. EDA features were the most significant, with `eda_ArgMax` being the single most important feature. Using the top four features, all from EDA, the IF method achieved the best AUC of 88.05% and an accuracy of 80.43%. Using the top ECG or EMG features resulted in low AUC values, around 60%. Surprisingly, the baseline of direct thresholding with `eda_ArgMax`

Table 6.2: Results of the methods that are trained on the hand-crafted features for cross-subject anomaly/pain detection. The Mean and Standard deviation (in %) received from a LOSO cross-validation evaluation are reported. Each method is trained on features from a single signal, while D represents the number of features used.

Method	D	Thresholding	Acc	Prec	Rec	AUC
IF (EDA)	4	85th-Percentile	80.43±14.02	82.54±13.31	75.86±26.37	**88.05±13.42**
		KARCH	79.17±13.10	78.23±13.33	**80.11±23.51**	
IF (ECG)	3	64th-Percentile	58.59±12.90	56.18±20.06	52.41±29.45	60.21±16.46
		KARCH	56.06±09.90	53.77±11.63	67.18±27.26	
IF (EMG)	1	84th-Percentile	57.79±12.74	52.18±38.19	31.49±34.85	60.53±18.28
		KARCH	55.98±12.10	52.38±26.59	50.75±37.77	
ArgMax (EDA)	1	90th-Percentile	**82.61±13.75**	**86.79±12.22**	75.63±25.49	84.18±14.42
		KARCH	80.78±13.39	81.26±11.97	78.74±23.99	
RF (EDA)	9	Supervised (0.5)	84.17±13.38	85.63±11.73	80.80±21.97	90.08±12.33

achieved the highest accuracy of 82.61% and the second-best AUC of 84.18%. As expected, RF as a supervised method achieved the best performance, though not by a significant margin.

Table 6.3: Results of the IF method for anomaly detection in a subject-specific 4-fold cross-validation evaluation: Mean and standard deviation (in %) for each metric across all subjects are reported. Each method is trained on features from a single signal, while D represents the number of features used.

Signals	D	Thresholding	Acc	Prec	Rec	AUC
EDA	8	84th-Percentile	**78.76±16.85**	**79.51±22.26**	**75.98±29.86**	**86.78±18.06**
		KARCH	72.93±18.77	65.77±34.66	66.21±38.78	
ECG	8	63th-Percentile	58.88±17.41	58.21±21.60	62.53±29.68	63.32±23.26
		KARCH	53.10±11.77	25.16±34.89	24.54±37.31	
EMG	7	84th-Percentile	56.90±18.13	55.13±22.88	59.08±30.42	61.67±23.54
		KARCH	54.74±13.99	38.75±3620	37.59±38.73	

In the subject-specific evaluation, IF again performed best on EDA features, achieving an AUC of 86.78% and an accuracy of 78.76%. The results are listed in Table 6.3. Notably, a greater number of features D was selected. For feature-based AD methods in general, percentile-based thresholds consistently outperformed the KARCH thresh-

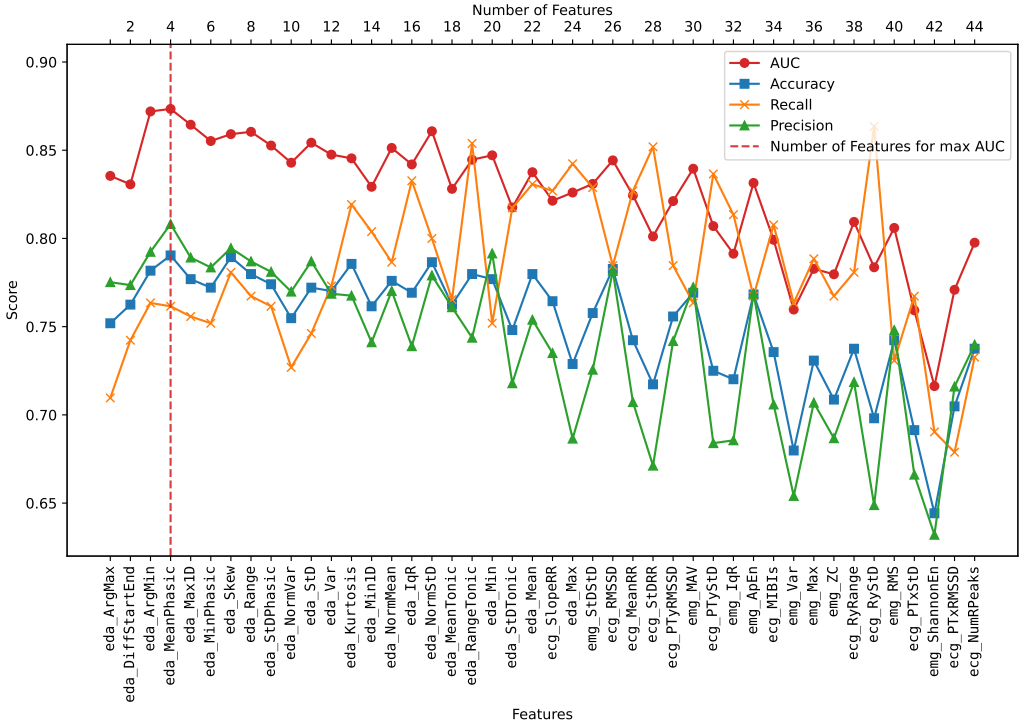
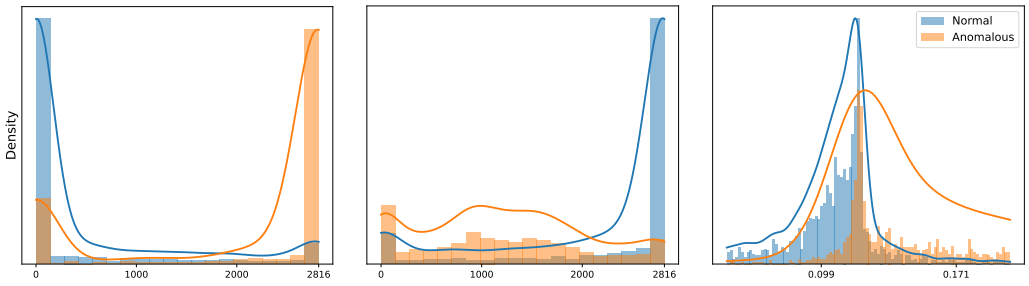


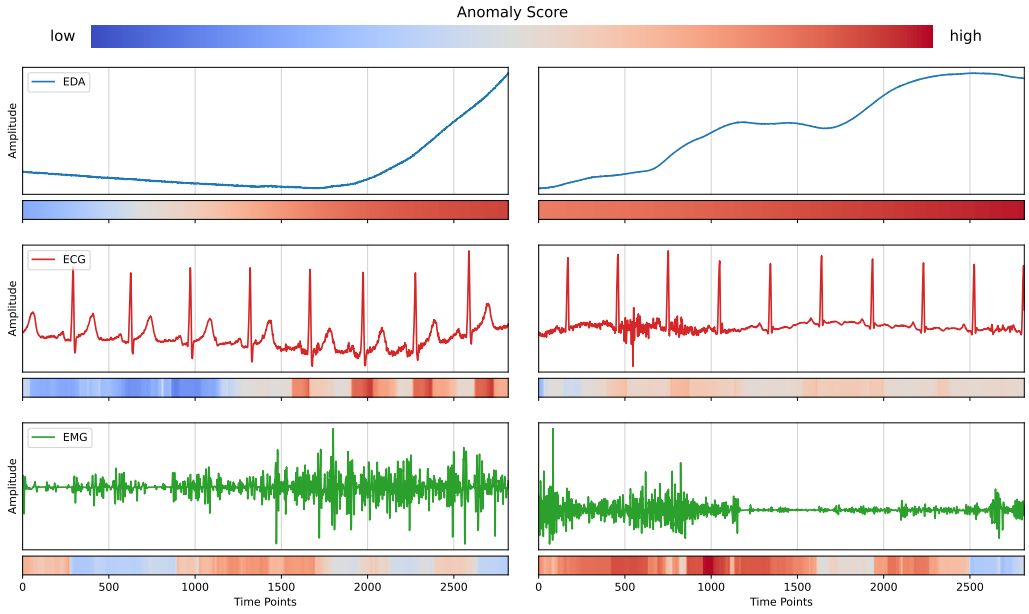
Figure 6.3: Performance of the IF using an increasing number of features. The features, as shown on the bottom x-axis, are ranked by importance (in descending order from left to right) using the RF classifier.



(a) `eda_ArgMax` (b) `eda_ArgMin` (c) `eda_DiffStartEnd`

Figure 6.4: Distributions of the three most important features for normal data (blue) and anomalous data (orange).

old. The selected percentile was higher (around 85) compared to the MP method (around 75).

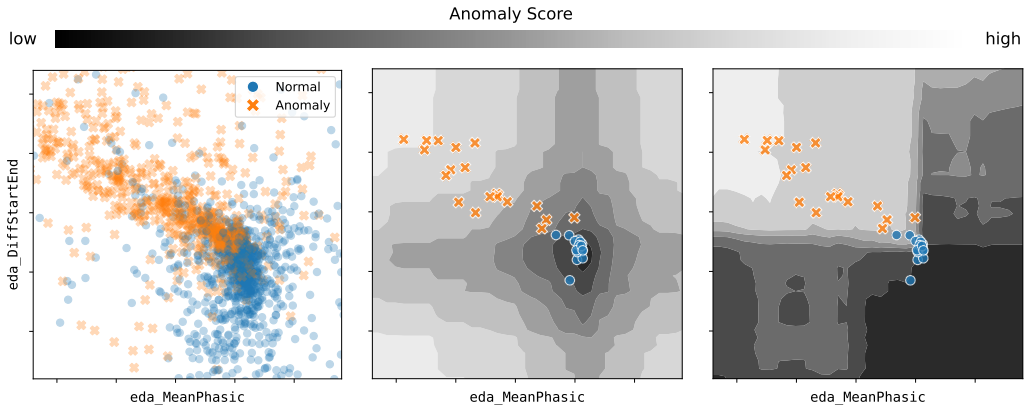


(a) 071309_w_21-PA4-068 (b) 072609_w_23-PA4-070

Figure 6.5: Matrix profile/point-wise anomaly scores calculated with the MP method for two random anomalous samples. The anomaly scores are visualized as a color gradient, where blue represents low scores, red represents high scores, and white is calibrated to the subject-dependent decision threshold. Each matrix profile is stretched to match the length of the original signal.

6.3 Autoencoder

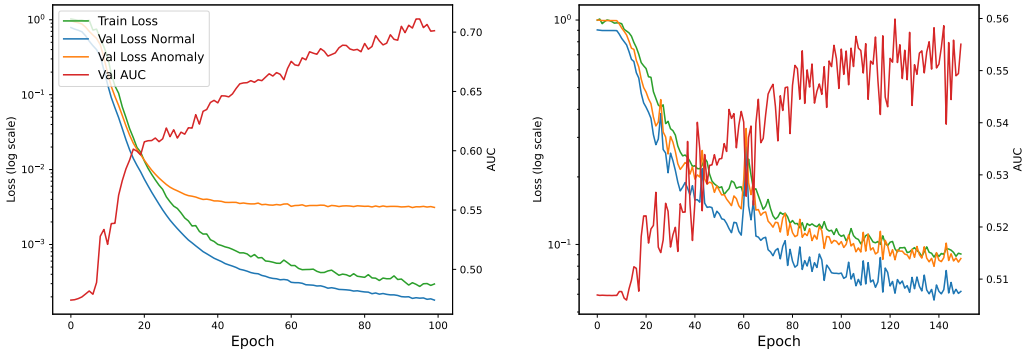
The performance of the AE was compared against the baseline model, PolyFit, for cross-subject AD. The results are shown in Table 6.4. The baseline model, PolyFit, applied to the EDA signal demonstrated the best performance among all methods, achieving an accuracy of 80.48%, a precision of 80.15%, and a recall of 82.18%, with an AUC score of 87.92%. The AE model applied to EDA signals achieved the best performance among AE models, with an AUC of 81.69%. However, compared to PolyFit, the AE model scored significantly lower, with an accuracy of 67.41%. For ECG and EMG, the AE model performed similarly to IF and MP, with an AUC around 60% and an accuracy around 55%. Combining all three signals did not lead to performance improvements. Figure 6.7 shows the declining loss curves and increasing



(a) Training data (b) Isolation Forest (c) Random Forest

Figure 6.6: Learning paradigm comparison: The training data (a) consists of labeled normal samples (blue) and anomalies (orange) in a 2D feature space (`eda_MeanPhasic` and `eda_DiffStartEnd`). (b) Isolation Forest decision boundaries, trained on normal samples only. (c) Random Forest decision boundaries, trained on both classes. The test samples are from a single test subject (`101609_m_36`).

AUC for each epoch during training. The growing separation between the validation loss of normal data and anomalous data aligns with expected behavior, where higher reconstruction loss for anomalies enables their detection.



(a) EDA Losses and AUCs (b) ECG Losses and AUCs

Figure 6.7: Losses and AUCs for the EDA (a) and ECG (b) autoencoder models: Training loss (green), validation loss of the normal data (blue), validation loss of the anomalous data (orange), and the validation AUCs (red).

Table 6.4: Results of the autoencoder and baseline model for cross-subject anomaly/pain detection. The Mean and Standard deviation (in %) received from a LOSO cross-validation evaluation are reported.

Method	Thresholding	Acc	Prec	Rec	AUC
PolyFit (EDA)	79th-Percentile	**80.48±12.80**	**80.15±12.68**	**82.18±23.33**	**87.92±13.67**
AE (EDA)	85th-Percentile	67.41±17.28	67.33±37.06	53.33±41.38	81.69±18.70
AE (ECG)	55th-Percentile	54.71±10.95	52.03±29.26	59.13±41.66	63.75±19.73
AE (EMG)	85th-Percentile	55.60±11.34	45.59±39.37	26.20±32.15	60.05±18.93
AE (EDA, ECG, EMG)	73th-Percentile	56.06±10.81	49.55±32.63	41.20±39.15	61.54±20.31

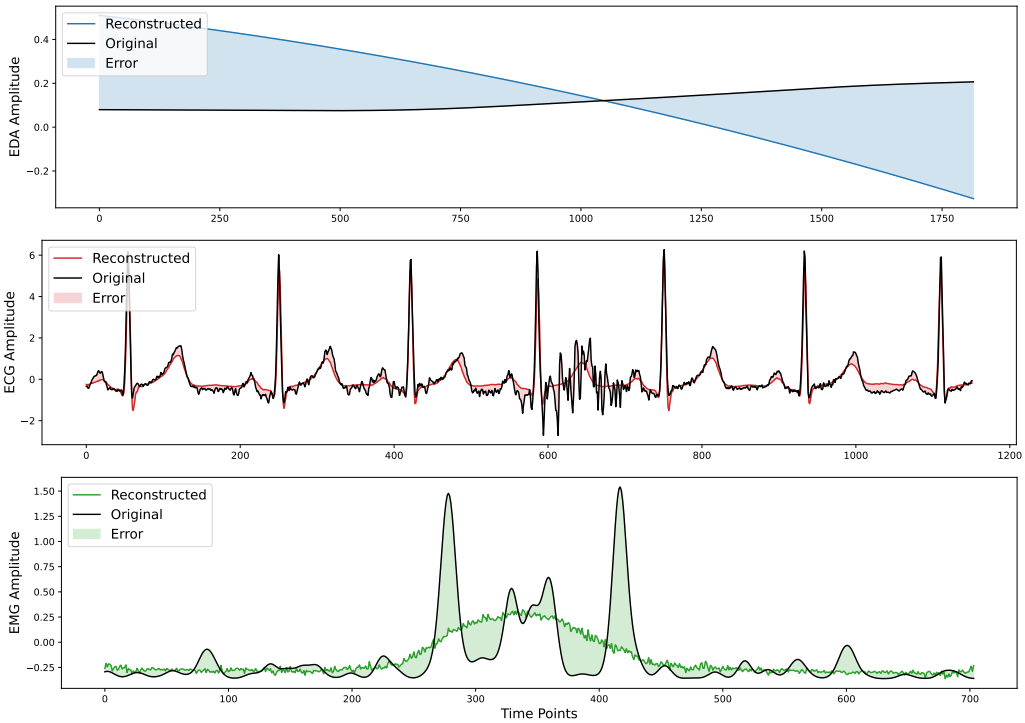


Figure 6.8: Reconstruction of EDA (blue), ECG (red), and EMG (green) signals from different pain labeled samples using PolyFit for EDA and the autoencoder for ECG and EMG.

7 Discussion

This chapter provides a detailed discussion of the results presented in the previous chapter. The experiments demonstrated that methods relying solely on the EDA signal outperformed those based on ECG, EMG, or a combination of signals. The EDA signal consistently provided the highest accuracy across all tested AD methods, aligning with findings from previous research [3, 23]. The tested AD methods performed worse than the supervised RF classifier, which is not surprising given that pain characteristics are expected to form clusters. This clustering is evident in the decision boundary plots (Figure 6.6) and the density estimations of the extracted features (Figure 6.2). Nonetheless, AD methods based on EDA achieved competitive accuracy (up to 82.61%), approaching state-of-the-art supervised methods (~85%) while only requiring normal data. Moreover, AD methods are capable of identifying any data points that deviate from the normal data distribution, as they model the normal space and treat deviations as potential anomalies. In contrast, supervised methods are limited to correctly detecting anomalies only if they fall within the learned boundaries of the anomalous class. This limitation arises because supervised methods explicitly learn a decision boundary between normal and anomalous classes, assuming these classes are well-represented and separable in the feature space. Because the characteristics of pain are expected to form distinct clusters, supervised methods should be preferred whenever sufficient labeled data is available.

The three methods achieving the highest AUC were: IF using the EDA feature with an AUC of 88.05%, PolyFit using the EDA feature with an AUC of 87.92%, and direct thresholding on the EDA feature ArgMax with an AUC of 84.18%. However, these methods relied on significant prior knowledge about the distribution of anomalies. The feature selection process for IF was informed by the feature importance ranking of the supervised RF classifier and prior research on pain detection.

In contrast, the MP method required minimal prior knowledge about the characteristics of pain-related anomalies and still achieved a competitive AUC of 83.73% using a combination of EDA and ECG signals. A notable disadvantage of the MP method is its susceptibility to the "twin freak" problem [36], where a single anomalous pattern

in the training or reference data can generate numerous false positives. One way to mitigate this issue, which showed no improvement in this work, is to consider the k-nearest neighbors rather than just the nearest neighbor [65]. A potentially more effective strategy to enhance the detection performance of the MP method could involve leveraging a human expert, similar to [32], to select a few reference samples that accurately represent normality.

The matrix profile method has the advantage of requiring only one main parameter, the window length. In contrast, other methods involve multiple parameters, making them more challenging to tune. The autoencoder, in particular, requires numerous hyperparameters to be optimized. Since most of these hyperparameters were set empirically, it is likely that the autoencoder's performance could be improved through systematic hyperparameter tuning, potentially using automated techniques in the future.

The MP method demonstrated the best performance when utilizing the ECG signal. However, unlike other methods, the ECG signal used in the MP approach was not detrended or preprocessed. This raises questions about whether the MP method's superior performance with ECG data was due to its robustness in capturing unique signal aspects or artifacts introduced by the EDA signal. As noted in [3], ECG signals exhibit baseline wandering that correlates strongly with the shape of the EDA signal. Despite detrending, minor artifacts remained in the ECG signal that could have been exploited by methods like IF and the autoencoder. This raises the question of whether more accurate detrending might further reduce the effectiveness of ECG signals for pain detection.

Analysis of the reconstructed EDA signals (Figure 6.8), the EDA matrix profile (Figure 4.1), histograms of EDA features, and individual EDA signals suggests that the EDA signal is primarily characterized by a linear trend. Normal EDA signals are generally constant or decreasing over time, with a maximum at the beginning of the series. In contrast, anomalous EDA signals tend to increase over time, with a maximum at the end of the series. Anomalous behavior in ECG and EMG signals is more difficult to interpret. However, the reconstruction of the EMG signals (Figure 6.8) suggests that a sudden increase in the trapezius muscle activity is correlated with the experience of heat pain. The ECG signals were the most challenging to interpret due to the high variance in ECG patterns across subjects. A future approach incorporating normal samples as context during inference might prove beneficial. The autoencoder, performed well in removing noise, baseline correction, and setting the same shape and mean to all cardiac cycles in an ECG signal. However, it was observed that the

autoencoder could not modify the lengths of intervals such as RR-intervals or PP-intervals. The matrix profile, by contrast, was better suited for capturing anomalies related to interval changes, while methods like Isolation Forest leveraged explicit features like RR and PT intervals for AD. In contrast, both the matrix profile method and isolation forest can capture anomalies related to interval changes.

Another consideration is the practical utility of each method in terms of computational efficiency. While all tested methods are feasible for real-time, online AD, notable differences exist in computational complexity. Isolation Forest and simple thresholding on a single feature are expected to be the fastest, whereas the autoencoder is relatively slower due to its architectural complexity. The computational demands of the MP method depend heavily on time series length, suggesting a need for future studies to evaluate its computational efficiency. Additionally, semi-supervised methods could be explored in future work, as they could combine the strengths of both supervised and unsupervised approaches.

The autoencoder based on 1D-convolutional layers did not benefit from combining multiple signals. Alternative architectures, such as 2D-convolutional layers, transformers, or recurrent neural networks, might better capture information across channels and model temporal dependencies more effectively.

Overall, ECG and EMG signals individually proved less effective for pain detection but could be valuable in developing a robust AD system that does not solely rely on the EDA signal. Since EDA responses are not specific to pain and may be triggered by other events, combining EDA with additional signals could enhance robustness. The challenge remains to develop a system that integrates multiple signals and possibly other modalities to improve reliability and performance.

8 Conclusion

This study evaluated three anomaly detection methods – Matrix Profile, Isolation Forest, and Autoencoder – alongside simpler approaches, PolyFit and ArgMax, for pain detection in biophysiological time series data from the BioVid Heat Pain Database (BVDB). These methods were compared against supervised classification approaches to assess their viability for this task.

Isolation Forest using EDA features achieved the highest AUC (88.05%) among anomaly detection methods, followed closely by PolyFit (87.92%) and ArgMax (84.18%). While these simpler methods required significant prior knowledge about anomaly distributions, Matrix Profile achieved a competitive AUC of 83.73% using minimal assumptions. The Autoencoder, despite its complexity, lagged behind with a best AUC of 81.69% and showed no improvement when combining multiple signals.

Supervised methods, such as the tested Random Forest classifier, outperformed anomaly detection methods, underscoring the advantage of leveraging labeled data. Pain anomalies were found to form distinct clusters, making supervised methods better suited for the task when sufficient labeled data is available. However, anomaly detection methods demonstrated a key strength: the ability to detect any deviation from normal data, not just those within the boundary of the pain class.

The experiments showed that EDA signals consistently provided the most predictive features for heat pain detection, aligning with prior research. ECG and EMG signals, while less effective individually, may be important for enhancing the robustness of multimodal systems.

In summary, while supervised methods remain superior for pain detection due to their ability to exploit labeled data, anomaly detection methods are effective in: 1) Detecting novel anomalies beyond pain-related anomalies, making them versatile for broader anomaly identification tasks; 2) Scenarios where no labeled anomalous data is available; and 3) Preprocessing steps, where they can potentially be used to clean data before applying supervised methods.

Bibliography

[1] Chitra Lalloo and Jennifer N Stinson. “Assessment and treatment of pain in children and adolescents”. In: *Best Practice & Research Clinical Rheumatology* 28.2 (2014), pp. 315–330.

[2] Steffen Walter et al. *Automatic pain quantification using autonomic parameters*. 2014.

[3] Patrick Thiam et al. “Exploring Deep Physiological Models for Nociceptive Pain Recognition”. In: *Sensors* 19.20 (2019), p. 4503. DOI: 10.3390/S19204503.

[4] Marco Cascella et al. “Artificial intelligence for automatic pain assessment: research methods and perspectives”. In: *Pain Research and Management* 2023.1 (2023), p. 6018736.

[5] Steffen Walter et al. “The biovid heat pain database data for the advancement and systematic validation of an automated pain recognition system”. In: *2013 IEEE International Conference on Cybernetics, CYBCO 2013, Lausanne, Switzerland, June 13-15, 2013*. IEEE, 2013, pp. 128–131. DOI: 10.1109/CYBCONF.2013.6617456.

[6] Renjie Wu and Eamonn J. Keogh. “Current Time Series Anomaly Detection Benchmarks are Flawed and are Creating the Illusion of Progress”. In: *IEEE Trans. Knowl. Data Eng.* 35.3 (2023), pp. 2421–2429. DOI: 10.1109/TKDE.2021.3112126.

[7] M. Saquib Sarfraz et al. “Position: Quo Vadis, Unsupervised Time Series Anomaly Detection?” In: *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[8] Lukas Ruff et al. “A Unifying Review of Deep and Shallow Anomaly Detection”. In: *Proc. IEEE* 109.5 (2021), pp. 756–795. DOI: 10.1109/JPROC.2021.3052449.

[9] Kukjin Choi et al. “Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines”. In: *IEEE Access* 9 (2021), pp. 120043–120065. DOI: 10.1109/ACCESS.2021.3107975.

[10] Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Anomaly detection: A survey”. In: *ACM Comput. Surv.* 41.3 (2009), 15:1–15:58. DOI: 10.1145/1541880.1541882.

[11] Kwei-Herng Lai et al. “Revisiting Time Series Outlier Detection: Definitions and Benchmarks”. In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*. Ed. by Joaquin Vanschoren and Sai-Kit Yeung. 2021.

[12] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. “Anomaly Detection in Time Series: A Comprehensive Evaluation”. In: *Proc. VLDB Endow.* 15.9 (2022), pp. 1779–1797. DOI: 10.14778/3538598.3538602.

[13] Zhenyu Zhong et al. “A Survey of Time Series Anomaly Detection Methods in the AIOps Domain”. In: *CoRR* abs/2308.00393 (2023). DOI: 10.48550/ARXIV.2308.00393.

[14] Lukas Ruff et al. “Deep Semi-Supervised Anomaly Detection”. In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[15] Guansong Pang et al. “Deep Weakly-supervised Anomaly Detection”. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*. Ed. by Ambuj K. Singh et al. ACM, 2023, pp. 1795–1807. DOI: 10.1145/3580305.3599302.

[16] Adrian Komadina et al. “Comparing Threshold Selection Methods for Network Anomaly Detection”. In: *IEEE Access* 12 (2024), pp. 124943–124973. DOI: 10.1109/ACCESS.2024.3452168.

[17] Md. Tahmid Rahman Laskar et al. “Extending Isolation Forest for Anomaly Detection in Big Data via K-Means”. In: *ACM Trans. Cyber Phys. Syst.* 5.4 (2021), 41:1–41:26. DOI: 10.1145/3460976.

[18] Willem van Veluw. “Application of Mixture Models to Threshold Anomaly Scores”. MA thesis. 2023.

[19] Matthew Middlehurst, Patrick Schäfer, and Anthony J. Bagnall. “Bake off redux: a review and experimental evaluation of recent time series classification algorithms”. In: *Data Min. Knowl. Discov.* 38.4 (2024), pp. 1958–2031. DOI: 10.1007/S10618-024-01022-1.

[20] Alejandro Pasos Ruiz et al. “The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances”. In: *Data Min. Knowl. Discov.* 35.2 (2021), pp. 401–449. DOI: 10.1007/S10618-020-00727-3.

[21] Angus Dempster, François Petitjean, and Geoffrey I. Webb. “ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels”. In: *Data Min. Knowl. Discov.* 34.5 (2020), pp. 1454–1495. DOI: 10.1007/S10618-020-00701-Z.

[22] Matthew Middlehurst et al. “HIVE-COTE 2.0: a new meta ensemble for time series classification”. In: *Mach. Learn.* 110.11 (2021), pp. 3211–3243. DOI: 10.1007/S10994-021-06057-9.

[23] Fatemeh Pouromran, Srinivasan Radhakrishnan, and Sagar Kamarthi. “Exploration of physiological sensors, features, and machine learning models for pain intensity estimation”. In: *Plos one* 16.7 (2021), e0254108.

[24] Philip Gouverneur et al. “Comparison of Feature Extraction Methods for Physiological Signals for Heat-Based Pain Recognition”. In: *Sensors* 21.14 (2021), p. 4838. DOI: 10.3390/S21144838.

[25] Zhenyuan Lu, Burcu Ozek, and Sagar V. Kamarthi. “Transformer Encoder with Multiscale Deep Learning for Pain Classification Using Physiological Signals”. In: *CoRR* abs/2303.06845 (2023). DOI: 10.48550/ARXIV.2303.06845.

[26] Kim Ngan Phan et al. “Pain Recognition With Physiological Signals Using Multi-Level Context Information”. In: *IEEE Access* 11 (2023), pp. 20114–20127. DOI: 10.1109/ACCESS.2023.3248654.

[27] Heng Shi, Belkacem Chikhaoui, and Shengrui Wang. “Tree-Based Models for Pain Detection from Biomedical Signals”. In: *Participative Urban Health and Healthy Aging in the Age of AI - 19th International Conference, ICOST 2022, Paris, France, June 27-30, 2022, Proceedings*. Ed. by Hamdi Aloulou et al. Vol. 13287. Lecture Notes in Computer Science. Springer, 2022, pp. 183–195. DOI: 10.1007/978-3-031-09593-1_14.

[28] Philip Gouverneur et al. “Explainable Artificial Intelligence (XAI) in Pain Research: Understanding the Role of Electrodermal Activity for Automated Pain Recognition”. In: *Sensors* 23.4 (2023), p. 1959. DOI: 10.3390/S23041959.

[29] Saranya Devi Subramaniam and Brindha Dass. “Automated nociceptive pain assessment using physiological signals and a hybrid deep learning network”. In: *IEEE Sensors Journal* 21.3 (2020), pp. 3335–3343.

[30] Patrick Thiam et al. “Multi-modal pain intensity assessment based on physiological signals: A deep learning perspective”. In: *Frontiers in Physiology* 12 (2021), p. 720464.

[31] Paul Boniol, John Paparrizos, and Themis Palpanas. “New Trends in Time Series Anomaly Detection.” In: *EDBT*. 2023, pp. 847–850.

[32] Matej Kloska, Gabriela Grmanová, and Viera Rozinajová. “Expert enhanced dynamic time warping based anomaly detection”. In: *Expert Syst. Appl.* 225 (2023), p. 120030. DOI: 10.1016/J.ESWA.2023.120030.

[33] Kieran Flanagan et al. “Network anomaly detection in time series using distance based outlier detection with cluster density analysis”. In: *2017 Internet Technologies and Applications (ITA)*. IEEE. 2017, pp. 116–121.

[34] Paul Boniol et al. “SAND: Streaming Subsequence Anomaly Detection”. In: *Proc. VLDB Endow.* 14.10 (2021), pp. 1717–1729. DOI: 10.14778/3467861.3467863.

[35] Chin-Chia Michael Yeh et al. “Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets”. In: *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain*. Ed. by Francesco Bonchi et al. IEEE Computer Society, 2016, pp. 1317–1322. DOI: 10.1109/ICDM.2016.0179.

[36] Yue Lu et al. “Matrix Profile XXIV: Scaling Time Series Anomaly Detection to Trillions of Datapoints and Ultra-fast Arriving Data Streams”. In: *KDD ’22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. Ed. by Aidong Zhang and Huzefa Rangwala. ACM, 2022, pp. 1173–1182. DOI: 10.1145/3534678.3539271.

[37] Kai Wang et al. “Research on Healthy Anomaly Detection Model Based on Deep Learning from Multiple Time-Series Physiological Signals”. In: *Sci. Program.* 2016 (2016), 5642856:1–5642856:9. DOI: 10.1155/2016/5642856.

[38] Edin Sabic et al. “Healthcare and anomaly detection: using machine learning to predict anomalies in heart rate data”. In: *AI Soc.* 36.1 (2021), pp. 149–158. DOI: 10.1007/S00146-020-00985-1.

[39] Markus M. Breunig et al. “LOF: Identifying Density-Based Local Outliers”. In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*. Ed. by Weidong Chen, Jeffrey F. Naughton, and Philip A. Bernstein. ACM, 2000, pp. 93–104. DOI: 10.1145/342009.335388.

[40] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. “Isolation Forest”. In: *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*. IEEE Computer Society, 2008, pp. 413–422. DOI: 10.1109/ICDM.2008.17.

[41] Miroslaw Staron et al. “Comparing Anomaly Detection and Classification Algorithms: A Case Study in Two Domains”. In: *Software Quality: Higher Software Quality through Zero Waste Development - 15th International Conference, SWQD 2023, Munich, Germany, May 23-25, 2023, Proceedings*. Ed. by Daniel Méndez et al. Vol. 472. Lecture Notes in Business Information Processing. Springer, 2023, pp. 121–136. DOI: 10.1007/978-3-031-31488-9_7.

[42] Hongzuo Xu et al. “Deep Isolation Forest for Anomaly Detection”. In: *IEEE Trans. Knowl. Data Eng.* 35.12 (2023), pp. 12591–12604. DOI: 10.1109/TKDE.2023.3270293.

[43] Zhihan Yue et al. “TS2Vec: Towards Universal Representation of Time Series”. In: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 2022, pp. 8980–8987. DOI: 10.1609/AAAI.V36I8.20881.

[44] Stéphane Crépey et al. “Anomaly Detection in Financial Time Series by Principal Component Analysis and Neural Networks”. In: *Algorithms* 15.10 (2022), p. 385. DOI: 10.3390/A15100385.

[45] Rushabh Musthyala, Shubham Arawkar, and Manik Gupta. “Anomaly identification using multimodal physiological signals on the edge”. In: *Proceedings of the 2nd Workshop on Deep Learning for Wellbeing Applications Leveraging Mobile Devices and Edge Computing*. 2021, pp. 7–12.

[46] Chunyong Yin et al. “Anomaly Detection Based on Convolutional Recurrent Autoencoder for IoT Time Series”. In: *IEEE Trans. Syst. Man Cybern. Syst.* 52.1 (2022), pp. 112–122. DOI: 10.1109/TSMC.2020.2968516.

[47] Bin Zhou et al. “BeatGAN: Anomalous Rhythm Detection using Adversarially Generated Time Series”. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by Sarit Kraus. ijcai.org, 2019, pp. 4433–4439. DOI: 10.24963/IJCAI.2019/616.

[48] Ioana Pintilie, Andrei Manolache, and Florin Brad. “Time Series Anomaly Detection using Diffusion-based Models”. In: *IEEE International Conference on Data Mining, ICDM 2023 - Workshops, Shanghai, China, December 4, 2023*. Ed. by Jihe Wang et al. IEEE, 2023, pp. 570–578. DOI: 10.1109/ICDMW60847.2023.00080.

[49] Yuhang Chen et al. “ImDiffusion: Imputed Diffusion Models for Multivariate Time Series Anomaly Detection”. In: *Proc. VLDB Endow.* 17.3 (2023), pp. 359–372.

[50] Daesoo Lee, Sara Malacarne, and Erlend Aune. “Explainable time series anomaly detection using masked latent generative modeling”. In: *Pattern Recognit.* 156 (2024), p. 110826. DOI: 10.1016/J.PATCOG.2024.110826.

[51] Viacheslav Kozitsin, Iurii Katser, and Dmitry Lakontsev. “Online forecasting and anomaly detection based on the ARIMA model”. In: *Applied Sciences* 11.7 (2021), p. 3194.

[52] Markus Thill et al. “Anomaly Detection in Electrocardiogram Readings with Stacked LSTM Networks”. In: *Proceedings of the 19th Conference Information Technologies - Applications and Theory (ITAT 2019), Hotel Zornička, Donovaly, Slovakia, September 20-24, 2019*. Ed. by Petra Barancíková et al. Vol. 2473. CEUR Workshop Proceedings. CEUR-WS.org, 2019, pp. 17–25.

[53] Abrar Alamr and Abdel Monim Artoli. “Unsupervised Transformer-Based Anomaly Detection in ECG Signals”. In: *Algorithms* 16.3 (2023), p. 152. DOI: 10.3390/A16030152.

[54] Yan Zhu et al. “Matrix Profile II: Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins”. In: *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain*. Ed. by Francesco Bonchi et al. IEEE Computer Society, 2016, pp. 739–748. DOI: 10.1109/ICDM.2016.0085.

[55] Sheng Zhong and Abdullah Mueen. “MASS: distance profile of a query over a time series”. In: *Data Min. Knowl. Discov.* 38.3 (2024), pp. 1466–1492. DOI: 10.1007/S10618-024-01005-2.

[56] Sahand Hariri, Matias Carrasco Kind, and Robert J. Brunner. “Extended Isolation Forest”. In: *IEEE Trans. Knowl. Data Eng.* 33.4 (2021), pp. 1479–1489. DOI: 10.1109/TKDE.2019.2947676.

[57] Luisa Luebke et al. “Objective Measurement of Subjective Pain Perception with Autonomic Body Reactions in Healthy Subjects and Chronic Back Pain Patients: An Experimental Heat Pain Study”. In: *Sensors* 23.19 (2023), p. 8231. DOI: 10.3390/S23198231.

[58] Markus Kächele et al. “Adaptive confidence learning for the personalization of pain intensity estimation systems”. In: *Evol. Syst.* 8.1 (2017), pp. 71–83. DOI: 10.1007/S12530-016-9158-4.

[59] Dor Bank, Noam Koenigstein, and Raja Giryes. “Autoencoders”. In: *CoRR* abs/2003.05991 (2020).

[60] Pierre Baldi. “Autoencoders, Unsupervised Learning, and Deep Architectures”. In: *Unsupervised and Transfer Learning - Workshop held at ICML 2011, Bellevue, Washington, USA, July 2, 2011*. Ed. by Isabelle Guyon et al. Vol. 27. JMLR Proceedings. JMLR.org, 2012, pp. 37–50.

[61] Dan Li et al. “Classification of ECG signals based on 1D convolution neural network”. In: *19th IEEE International Conference on e-Health Networking, Applications and Services, Healthcom 2017, Dalian, China, October 12-15, 2017*. IEEE, 2017, pp. 1–6. DOI: 10.1109/HEALTHCOM.2017.8210784.

[62] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015.

[63] Vít Skvára, Tomás Pevný, and Václav Smídl. “Is AUC the best measure for practical comparison of anomaly detectors?” In: *CoRR* abs/2305.04754 (2023). DOI: 10.48550/ARXIV.2305.04754.

[64] Andrew P. Bradley. “The use of the area under the ROC curve in the evaluation of machine learning algorithms”. In: *Pattern Recognit.* 30.7 (1997), pp. 1145–1159. DOI: 10.1016/S0031-3203(96)00142-2.

[65] Chin-Chia Michael Yeh et al. “Matrix Profile for Anomaly Detection on Multidimensional Time Series”. In: *CoRR* abs/2409.09298 (2024). DOI: 10.48550/ARXIV.2409.09298.

Abbreviations

AD	Anomaly Detection
$AdaBoost$	Adaptive Boosting
AE	Autoencoder
$ARIMA$	AutoRegressive Integrated Moving Average
AUC	Area Under the Curve
$BVDB$	BioVid Heat Pain Database
CNN	Convolutional Neural Network
$Conv1d$	1D Convolutional Layer
$ConvTranspose1d$	1D Transposed Convolutional Layer
$DDCAE$	Deep Denoising Convolutional Autoencoder
DIF	Deep Isolation Forest
DTW	Dynamic Time Warping
EB	Elliptical Boundaries
ECG	Electrocardiogram
EDA	Electrodermal Activity
EM	Expectation-Maximization
EMG	Electromyogram
FFT	Fast Fourier Transform
$FILTER$	Filtering Method
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GSR	Galvanic Skin Response
IQR	Interquartile Range

IF	Isolation Forest
$iTree$	Isolation Tree
KNN	K-Nearest Neighbors
$KARCH$	Karcher mean (Riemannian Center of Mass)
LOF	Local Outlier Factor
$LOSO-CV$	Leave-One-Subject-Out Cross-Validation
$MIXMOD$	Normal and Non-Normal Mixture Models
MLE	Maximum Likelihood Estimation
MP	Matrix Profile
MSE	Mean Squared Error
NRS	Numerical Rating Scale
PCA	Principal Component Analysis
$PolyFit$	Polynomial Fit
$ReLU$	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristics
SCL	Skin Conductance Level
$STAMP$	Scalable Time Series Anytime Matrix Profile
$STOMP$	Scalable Time Series Ordered-Search Matrix Profile
SVM	Support Vector Machine
SVR	Support Vector Regression
TN	True Negative
TP	True Positive
TPR	True Positive Rate
VAS	Visual Analog Scale
$VQ-VAE$	Vector Quantized Variational Autoencoder
$XGBoost$	Extreme Gradient Boosting