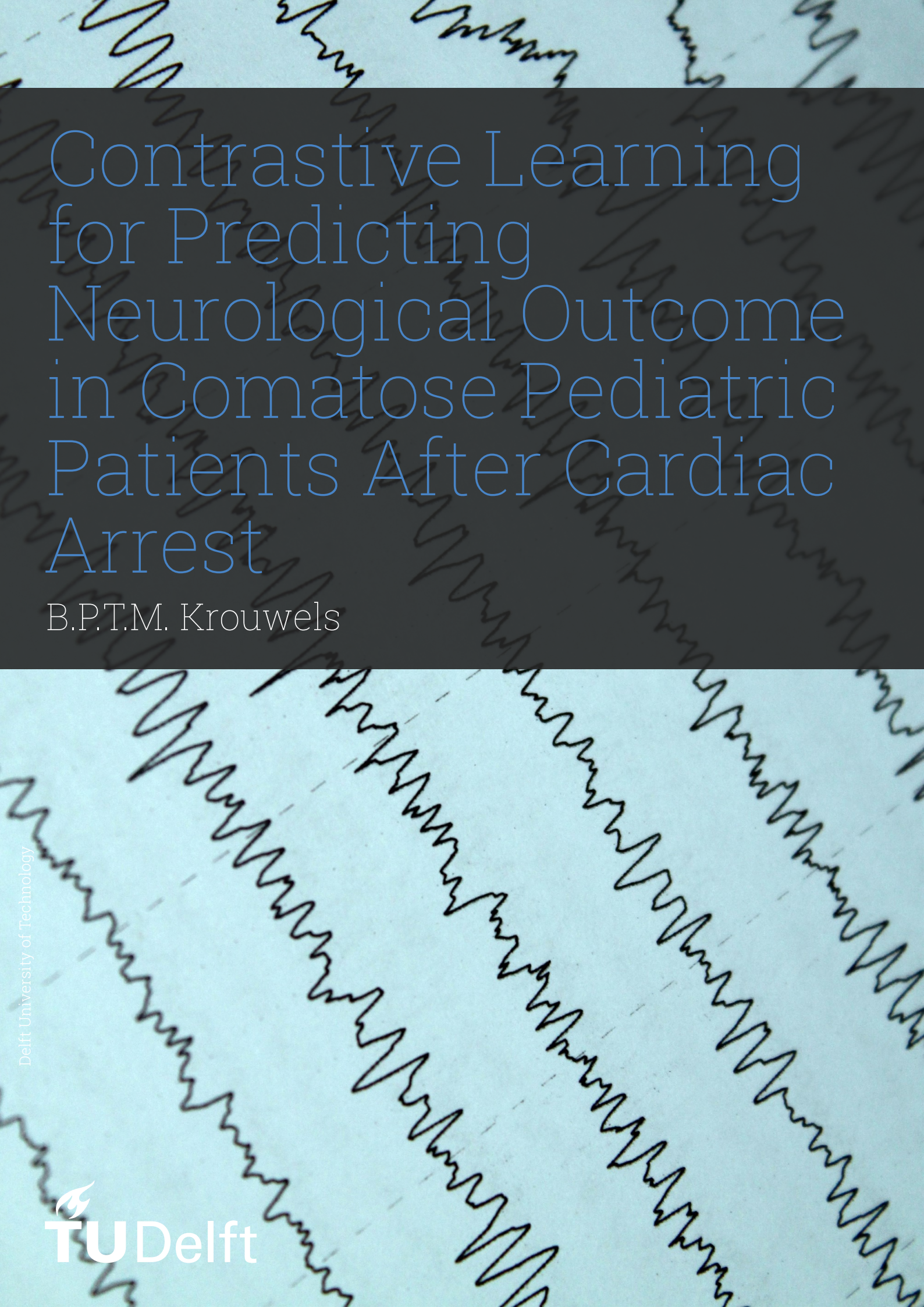


Contrastive Learning for Predicting Neurological Outcome in Comatose Pediatric Patients After Cardiac Arrest

B.P.T.M. Krouwels

Delft University of Technology

TU Delft

Contrastive Learning for Predicting Neurological Outcome in Comatose Pediatric Patients After Cardiac Arrest

by

B.P.T.M. Krouwels

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday June 27th, 2025 at 01:00 PM.

Student number:	4921909	
Project duration:	September, 2024 – June, 2025	
Thesis committee:	Prof. dr. ir. G. Jongbloed,	TU Delft, supervisor
	Dr. R. van den Berg,	Erasmus MC, supervisor
	Dr. ir. G. F. Nane,	TU Delft, thesis committee

Cover: The Medicine Study Electrophysiology by NomeVisualizzato [46].

TU Delft

Erasmus MC
Universitair Medisch Centrum Rotterdam

Preface

This thesis marks the completion of my Master's degree in Applied Mathematics, with a specialization in Mathematical Data Science at TU Delft. I feel incredibly fortunate to have concluded my studies with a project that combines mathematical modeling, machine learning, and neuroscience in a meaningful clinical context.

Working on a real-world challenge, predicting neurological outcomes in pediatric patients after cardiac arrest, has been both intellectually rewarding and personally motivating. This project has shown me how abstract mathematical tools can be translated into practical solutions that matter, especially in sensitive and high-stakes environments like in healthcare.

Throughout the past months, I have deepened my understanding of data-driven modeling, neural network architectures, and working in an interdisciplinary setting. I am grateful for the opportunity to explore such an interdisciplinary topic, and for the chance to contribute, in however small a way, to a problem of societal and medical importance. [1]

I would like to express my sincere gratitude to my thesis supervisors, Geurt Jongbloed and Robert van den Berg, for their invaluable guidance and encouragement throughout this project. Your enthusiasm and insightful feedback have been instrumental in keeping me motivated and engaged from start to finish. The interdisciplinary collaboration between TU Delft and Erasmus MC has been a particularly inspiring aspect of this project. It is remarkable to see how mathematics, data science, and clinical expertise can come together to tackle complex, real-world problems. My gratitude also goes out to Marit Verboom, a PhD candidate at Erasmus MC, who has supported me greatly through our weekly meetings, both personally and professionally. I would also like to thank Tina Nane for serving on my thesis committee and for your thoughtful input and time.

Finally, I wish to thank my parents for their continuous support during my seven years at TU Delft. Your encouragement has been a constant source of strength and motivation. I am also deeply grateful to my brother, sisters, and friends for their continued support, both during the fun moments that provided much-needed balance, and in the thoughtful conversations and practical help that contributed to this thesis. Your presence made this journey not only possible but also truly enjoyable.

B.P.T.M. Krouwels
Rotterdam, June 2025

[1] To improve grammar and writing flow, I used ChatGPT (OpenAI) as a language assistant. All scientific content, mathematical derivations, and conclusions were developed independently.

Abstract

Accurate early prognostication after pediatric cardiac arrest is clinically crucial yet remains methodologically challenging. This thesis frames the problem as a representation learning task on raw electroencephalography (EEG) and introduces a fully self-supervised pipeline followed by a supervised classifier. A Time-Series-to-Vector (TS2Vec) encoder is trained on 6,669 unlabeled twenty-second EEG epochs ($T = 2000$ time points, $C = 4$ channels) from 84 comatose children treated at Erasmus MC Sophia Children's Hospital. The encoder learns to map each EEG epoch to a 320-dimensional vector by contrasting transformed views of the same signal from unrelated signals, without using outcome labels.

Downstream classification is performed with a k-nearest neighbors (k-NN) model. Probabilities are assigned at the epoch level and aggregated across all epochs of a patient to generate a final patient-level prognosis. Five-fold patient-level cross-validation gives an area under the ROC curve of 0.861 ± 0.092, an accuracy of 0.774 ± 0.045 and an F1 score of 0.733 ± 0.057. Critically, specificity and precision are both 1.000 ± 0.000. Thus the model never predicts death for a child who survives, satisfying a stringent clinical safety requirement. Sensitivity is 0.581 ± 0.070, reflecting a deliberately conservative decision threshold.

Qualitative analyses offers tentative support for the physiological relevance of the learned features. t-SNE projections show clustering by patient identity and good separation between the two extreme EEG background patterns (continuous normal activity and electrocerebral silence) without supervision. Saliency analysis did not yield clinically interpretable patterns.

Compared to the leading feature-engineered qEEG baseline, the proposed approach achieves equivalent specificity and a slightly lower AUC, while removing the need for manual feature design and expert annotation. The modular architecture invites several extensions, including graph-based encoders that respect electrode topology, integration of auxiliary data such as ECG and clinical metadata, retrieval-based reasoning using large historical EEG archives, and more interpretable or end-to-end trainable models. Overall, this work demonstrates that contrastive learning can be effectively applied to raw pediatric EEG for conservative, label-efficient outcome prediction after cardiac arrest, and offers a mathematically grounded proof of concept for future clinical applications.

Contents

List of Figures

List of Tables

Abbreviations

Abbreviation	Definition
1D	One-dimensional
2D	Two-dimensional
3D	Three-dimensional
ACNS	American Clinical Neurophysiology Society
AUC	Area Under the (ROC) Curve
CNN	Convolutional Neural Network
ECG	Electrocardiogram
EEG	Electroencephalogram
GELU	Gaussian Error Linear Unit
k-NN	k-Nearest Neighbors
PCPC	Pediatric Cerebral Performance Category
qEEG	Quantitative Electroencephalogram
ReLU	Rectified Linear Unit
ROC	Receiver Operating Characteristic
ROSC	Return of Spontaneous Circulation
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
TS2Vec	Time Series to Vector
t-SNE	t-distributed Stochastic Neighbor Embedding

1 Introduction

Reliable early prediction of long-term neurological outcomes after pediatric cardiac arrest remains a critical, unsolved problem at the intersection of biomedical signal processing, clinical neurophysiology, and modern machine learning.

In this thesis, we develop and evaluate a self-supervised contrastive learning framework for raw EEG time series of comatose children after cardiac arrest. The goal is to enable early and reliable prediction of long-term neurological outcome, providing tools that may ultimately assist clinicians and families facing critical decisions under great uncertainty.

1.1. Motivation and Research Objectives

Predicting long-term neurological outcomes after cardiac arrest is a central challenge in pediatric intensive care. Electroencephalography (EEG) is a widely used tool for this purpose in adults, offering noninvasive, real-time insights into brain function [37]. For children however, this method has not been sufficiently validated for this indication. Traditionally, EEG-based outcome prediction has relied on handcrafted quantitative EEG (qEEG) features or expert-reviewed visual scores, which can be limited by domain assumptions and high annotation costs.

Recent advances in deep learning have shown promise for extracting complex, task-relevant features directly from raw medical time series data, bypassing the need for manual feature engineering [52][32]. In particular, self-supervised contrastive learning offers a compelling alternative to supervised approaches [29]. It can learn useful representations from unlabeled data by exploiting structural properties of the input itself. This is particularly relevant in clinical EEG, where high-quality labels are expensive and ambiguous to obtain. Raw EEG signals are high-dimensional, temporally complex, and often contaminated by artifacts, which makes representation learning particularly challenging and important.

While previous work has explored qEEG features for pediatric cardiac arrest [14][19], the potential of deep learning methods, such as contrastive learning, on raw EEG data has not yet been evaluated in this specific clinical setting [50]. This thesis aims to bridge that gap.

The study is guided by two central research objectives:

1. To investigate whether contrastive learning can extract clinically meaningful representations from raw EEG recordings of comatose pediatric patients after cardiac arrest.
2. To evaluate whether these learned representations can be used, via a downstream classifier, to predict long-term survival based on the Pediatric Cerebral Performance Category (PCPC) score at 12 months post-arrest.

More broadly, the learned representations may have potential utility in clinical decision support, for example by enabling comparison of a new patient to similar historical patients based on their EEG recordings, but this application is beyond the scope of the present study.

1.2. Clinical Background: EEG Monitoring after Pediatric Cardiac Arrest

Children who suffer cardiac arrest are at high risk of long-term neurological injury, even after return of spontaneous circulation (ROSC) is achieved. Accurate early prognosis is essential for guiding treatment decisions and counseling families. However, predicting long-term neurological outcome remains a major clinical challenge, particularly in pediatric populations [44].

EEG is one of the few tools available for real-time assessment of brain function in the intensive care setting. When used in the early days after cardiac arrest, EEG can help detect severe brain injury and guide prognostic assessment [41][40]. Traditionally, clinical EEG interpretation relies on visual analysis by trained neurophysiologists, but this process is time consuming and may suffer from inter-rater variability.

To address this, researchers have explored automated EEG analysis using qEEG features such as signal amplitude, spectral power, and continuity metrics [14][19]. These qEEG features can be fed into machine learning classifiers to estimate neurological outcome, often with reasonable success. However, this approach has limitations: handcrafted features may omit clinically relevant dynamics, and feature design often encodes strong assumptions about what aspects of the EEG are informative.

In contrast, deep learning approaches can learn feature representations directly from raw EEG time series, enabling discovery of complex patterns that are difficult to capture with predefined descriptors. This opens the possibility for models that adapt to the full richness of the data without requiring expert-crafted input. To the best of our knowledge, based on literature published up to June 2025, this is the first study to apply self-supervised contrastive learning to raw pediatric EEG for long-term outcome prediction after cardiac arrest[50].

1.3. Outline of the Approach

This project introduces a deep learning pipeline for representation learning on pediatric EEG using self-supervised contrastive learning. The central idea is to train a neural encoder to extract informative features from raw, unlabeled EEG segments by solving a surrogate task that does not rely on outcome labels. Specifically, we adopt the Time Series to Vector (TS2Vec) framework [53], which uses timestamp masking and random cropping to construct positive training pairs, and optimizes a hierarchical contrastive loss to enforce invariance and contextual consistency in the learned representations. The details of our adaptation and implementation of TS2Vec are provided in Section 3.3.

Before training the encoder, the raw EEG recordings undergo a series of preprocessing steps to ensure consistency and quality. This includes artifact removal, downsampling, channel selection, and per-channel normalization, resulting in standardized 20-second EEG epochs with four channels. A full description of the preprocessing can be found in Section 2.3.

Once trained, the encoder is used to map unseen EEG epochs into a low-dimensional latent space. These representations, which aim to capture temporal and spectral patterns relevant to brain state, are then used as features for classification. We apply a k-nearest neighbors (k-NN) classifier directly on the latent embeddings to predict binary outcome labels, survival ($y = 0$) vs. death ($y = 1$) at 12 months, based on the PCPC score. Classification is first performed at the level of individual EEG epochs. To arrive at a final prediction per patient, the epoch-level outputs of the classifier are aggregated. This step reflects the clinical reality that outcome is assigned per patient rather than per EEG segment, and allows robust aggregation of the model's confidence across multiple time windows. This process is explained in Section 4.2.

In Section 5, the learned representations and the classifier's performance are evaluated using both qualitative and quantitative methods. To qualitatively assess the structure of the learned representations, we visualize the structure of the latent space using t-distributed Stochastic Neighbor Embedding (t-SNE) to assess clustering by outcome and clinical labels in Section 5.1. We also report standard classification metrics at the patient level, including AUC, accuracy, specificity, and precision in Section 5.2. These results demonstrate that the proposed contrastive learning pipeline can achieve good and conservative predictive performance in this clinically sensitive setting.

The final conclusions are presented in Section 6. These findings are discussed in Section 7, here also suggestions for future research are outlined.

The contributions of this thesis are both methodological and practical. A clean and standardized dataset was constructed by removing overlapping EEG segments, applying uniform resampling to 100 Hz, band-pass filtering (0.5–35 Hz), and channel-wise z-score normalization. A minimal yet physiologically balanced subset of channels (C3, C4, F7/Fp1, F8/Fp2) was selected to enable inclusion of all 84 pediatric patients. The TS2Vec framework was adapted for hierarchical (EEG) data and the encoder was trained on unlabeled 20-second EEG windows. For downstream classification, a k-nearest neighbors classifier was implemented to operate directly on the learned representations, along with a patient-level aggregation strategy based on a high posterior threshold to ensure conservative clinical predictions. To gain insight into the learned features and encoder behavior, both t-SNE visualizations and a saliency analysis pipeline were developed. All code, including preprocessing scripts, training procedures, classification scripts, and evaluation tools, has been made publicly available to support reproducibility. All methods in this study were implemented in Python using PyTorch. The full code is available at [1].

[1] https://github.com/BerendKr/Contrastive_Learning_Classification_on_EEG_Data

2

Data

High-quality data are crucial for developing effective machine learning models, especially in clinical applications where data can be complex, heterogeneous, and limited in size. In this study, we use EEG data collected from pediatric patients following cardiac arrest, sourced from an observational cohort study conducted at the Erasmus MC Sophia Children's Hospital in Rotterdam, the Netherlands [19]. The patient group consists of children between 0 and 17 years of age who suffered either in-hospital or out-of-hospital cardiac arrest and subsequently achieved return of spontaneous circulation (ROSC). EEG monitoring was performed to assess brain activity and neurological status post-resuscitation.

This dataset has been previously used for studies employing qEEG analysis to predict neurological outcomes [19], where features such as amplitude, frequency, and signal continuity were extracted before classification. In contrast, this thesis directly utilizes the raw EEG time series data, allowing for the capture of unexpected temporal dynamics that may be lost in pre-defined feature extraction. By working with raw data, we aim to leverage the full potential of neural networks and contrastive learning methods to discover complex patterns in the EEG signals associated with patient outcomes.

Ethical approval for the use of this dataset was obtained from the Erasmus MC Ethical Review Board (MEC-2019-0259 and MEC-2021-0145). Given the retrospective nature of the study and the use of anonymized clinical data, the need for informed consent was waived.

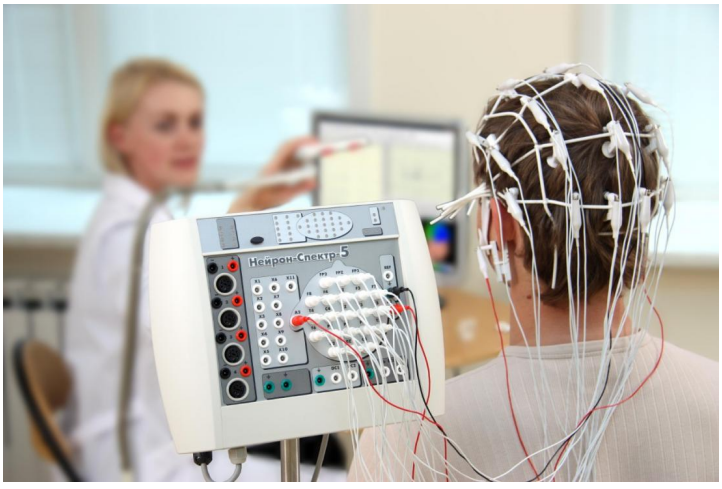


Figure 2.1: Example of EEG signals being captured by attaching electrodes to a patients head [3]

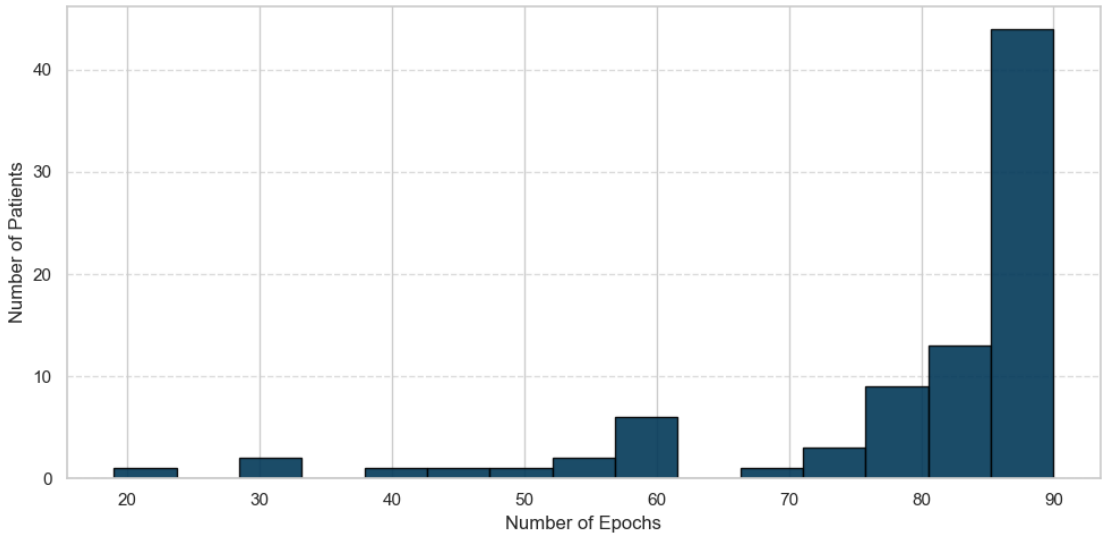


Figure 2.2: Histogram of the number of EEG epochs per patient.

2.1. Origin and Clinical Context

This dataset originates from the same clinical EEG dataset as used by Hunfeld et al. [19]. Below, we summarize the clinical and technical context in which the EEG data are collected.

The EEG recordings were made between January 2012 and November 2019 as part of routine clinical care. EEG monitoring was initiated approximately 24 hours after ROSC. Electrode placement followed the international 10-20 system, with the number of electrodes ranging from 11 to 19 depending on the patient's head circumference. The EEG signals were recorded using an OSG BrainRT system at a sampling frequency of 256 Hz. For the purposes of this study, the data were resampled to 100 Hz. This sampling rate adequately captures the frequency components of interest (up to 35 Hz) pertinent to background EEG analysis [37], while also reducing computational load and facilitating more efficient model training. Initial cleaning of the EEG recordings, including artifact removal and selection of usable epochs, was performed as part of the preprocessing pipeline described in Hunfeld et al. [19]. Subsequent steps, including normalization, resampling, filtering and channel selection, were conducted as part of this thesis and will be detailed in Section 2.3.

2.2. Data Characteristics

Each patient in the dataset has one continuous EEG recording that was split into multiple 20-second epochs during preprocessing. Given the uniform sampling rate of 100 Hz, each epoch contains $T = 2000$ time steps. With $C = 4$ channels selected, the raw EEG input per epoch is represented as a matrix $\boldsymbol{x}_i$ with dimensions $(T, C) = (2000, 4)$.

The dataset has a hierarchical structure, where multiple epochs are nested within each patient. After preprocessing, a total of 6,669 epochs are retained across 84 patients (totaling to 37 hours of EEG data). Each patient has between 19 and 89 epochs, the distribution is shown in Figure 2.2. On average, there are 79 epochs per patient.

This hierarchical structure has implications for modeling and evaluation. One expects epochs of the same patient to be more similar than those of different patients, therefore train-test splitting must be conducted at the patient level to prevent information leakage. This structure also informs our modeling setup where representations are learned at the epoch level, but classification is performed at the patient level by aggregating the epoch predictions per patient. To make sure that each patient is in the test set once, we used 5 fold cross validation. For each fold, 80% of the patients were used for training, 20% were used for testing.

2.2.1. Raw EEG Data

Each EEG segment is a 20-second recording sampled at 100 Hz, resulting in 2000 time points per epoch. After channel selection (see Section 2.3.1), each segment contains 4 channels, giving an input matrix $\boldsymbol{x}_i \in \mathbb{R}^{2000\times 4}$ per epoch. This format preserves the full temporal structure of the signal, including amplitude fluctuations and rhythm patterns. Before normalization, the amplitudes typically ranged between 20-100 μV, after normalization, each channel has mean 0 and variance 1. This per-channel normalization is important for training stability, particularly in convolutional neural networks, as it ensures that each input channel is scaled consistently, preventing any one channel from dominating the learning process due to differences in amplitude or variability. The normalization is computed over the entire dataset on a per-channel basis, so the relative amplitude proportions between epochs within the same channel are preserved.

Figure 2.3 illustrates a raw EEG epoch (before normalization) from a single patient using the four selected channels (C3, C4, F7, F8). As seen in the figure, the signal demonstrates variability in amplitude and frequency content typical of resting-state brain activity. EEG recordings from comatose patients tend to show reduced variability over time compared to those from awake individuals, as brain activity in coma is typically less responsive to external stimuli. Furthermore, since the patients in this dataset were resuscitated after cardiac arrest, their brains experienced a period of oxygen deprivation (hypoxia). This hypoxic insult can cause brain damage, disrupt neuronal activity, and result in reduced variability in the EEG [35].

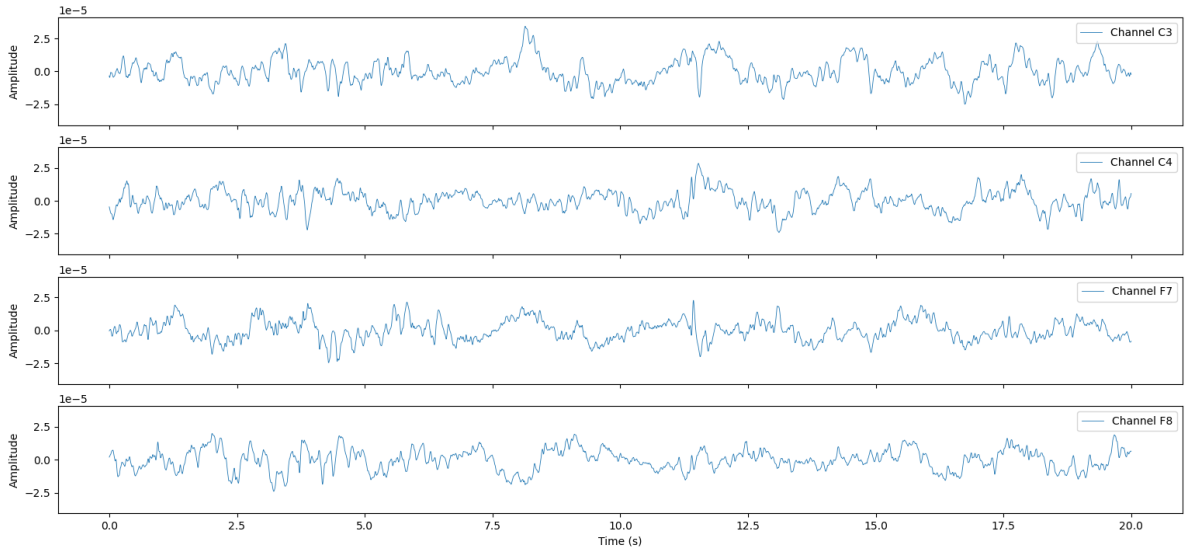


Figure 2.3: Example of a 20-second raw EEG epoch from one patient, showing the selected channels (C3, C4, F7, F8). This patient had a PCPC score of 1 after 12 months.

Our use of raw EEG contrasts with traditional approaches that rely on qEEG features. In prior work, EEG recordings were first reduced to a set of handcrafted features (e.g. power in specific frequency bands, burst suppression ratios or overall signal continuity) before being used in classification models [19][27][2][36]. While such qEEG features can offer interpretability, they require strong domain assumptions [23], and introduce the risk of discarding signal components that may carry predictive value. In contrast, our approach allows a neural encoder to model the full temporal structure of the EEG signal directly, without constraining it to predefined feature sets. This enables the extraction of subtle, potentially clinically relevant patterns from the raw data. As contrastive learning methods are particularly well suited for identifying latent structure in unannotated time series [31], the raw data format maximizes the model's capacity to learn rich representations that have predictive power for patient outcomes.

2.3. Data Preprocessing

The preprocessing pipeline consists of both previously published cleaning steps and additional transformations performed in this thesis. Initial cleaning was conducted by Hunfeld et al. and involves artifact rejection, noisy channel removal, and the segmentation of raw EEG signals into epochs of 20 seconds with 10 seconds of overlap. In this step, 2 patients were excluded from further analysis due to excessive artifact contamination across their entire recordings.

To ensure that all data are only seen once per training iteration, we removed the overlap by discarding every other epoch. This reduced the dataset to $n = 6,669$ non-overlapping epochs. All epochs were then resampled to a consistent sampling rate of 100 Hz using the the MNE-Python library [16]. A high-pass filter of 0.5 Hz and a low-pass filter of 35 Hz were applied to remove noise and to harmonize recordings that had already undergone partial filtering during acquisition. These filters setting are standard practice in EEG preprocessing to attenuate slow drifts (e.g. from perspiration or electrode movement) and high-frequency noise (e.g. muscle artifacts), thereby preserving the frequency bands most relevant for clinical EEG interpretation. Specifically, the 0.5 Hz high-pass filter effectively removes low-frequency artifacts, while the 35 Hz low-pass filter attenuates high-frequency noise without significantly affecting the EEG signals of interest. Each epoch $\boldsymbol{x}_i \in \mathbb{R}^{T \times C}$ was then normalized per channel using z-score normalization, implemented via the the StandardScaler from scikit-learn [34]. For clarity, we rewrite $\boldsymbol{x}_i$ as a tuple of C channel vectors of length T: $\boldsymbol{x}_i = (\boldsymbol{x}_i^{(1)}, \boldsymbol{x}_i^{(2)}, \ldots, \boldsymbol{x}_i^{(C)})$. Now, for each channel $c \in \{1, \ldots, C\}$, we compute the empirical mean $\mu_i^{(c)}$ and standard deviation $\sigma_i^{(c)}$:

$$\mu_i^{(c)} = \frac{1}{T} \sum_{t=1}^{T} x_{i,t}^{(c)}, \qquad \sigma_i^{(c)} = \sqrt{\frac{1}{T} \sum_{t=1}^{T} \left(x_{i,t}^{(c)} - \mu_i^{(c)} \right)^2}. \tag{2.1}$$

Then we use these values to normalize the $\boldsymbol{x}_i$ per channel:

$$\tilde{\boldsymbol{x}}_{i,t}^{(c)} = \frac{\boldsymbol{x}_{i,t}^{(c)} - \mu_i^{(c)}}{\sigma_i^{(c)}}, \quad \text{for } t = 1, \ldots, T. \tag{2.2}$$

The standardization ensures that each channel has empirical mean zero and unit variance. This prevents channels with larger amplitudes from disproportionately influencing the optimization process of the encoder, stabilizing training and improving convergence of the model [5].

While normalization removes absolute amplitude information from the input signals, which carries clinical relevance (e.g. lower amplitudes in severely injured brains), this loss is mitigated in our setup. The normalization is applied per channel across the entire dataset, meaning that inter-patient differences in amplitude scaling are preserved. A globally flatter EEG recording still results in a lower normalized variation across time compared to a highly dynamic recording. Only the absolute magnitude is adjusted and the mean is shifted. Thus, while some information loss is inherent, the relative structure of the EEG dynamics is maintained.

Nevertheless, it is important to note that this preprocessing choice implicitly assumes that relative rather than absolute EEG amplitudes are most informative for the downstream prediction task.

2.3.1. Channel Selection

While 11 to 19 channels were recorded, using all channels is often unnecessary in this specific patient population. In comatose patients with postanoxic encephalopathy, brain activity tends to be diffusely abnormal, leading to possible redundancy between spatially adjacent electrodes. Verboom investigated the effect of reducing the number of electrodes from 12 down to 4, and found that the performance of random forest models remained stable across this range [48]. This suggests that a smaller subset of electrodes may contain sufficient information for the model to make meaningful predictions. This is consistent with studies on visual EEG assessment, which suggest that outcome prediction largely depends on globally observable abnormalities in brain activity, rather than on localized signal characteristics [4] [1].

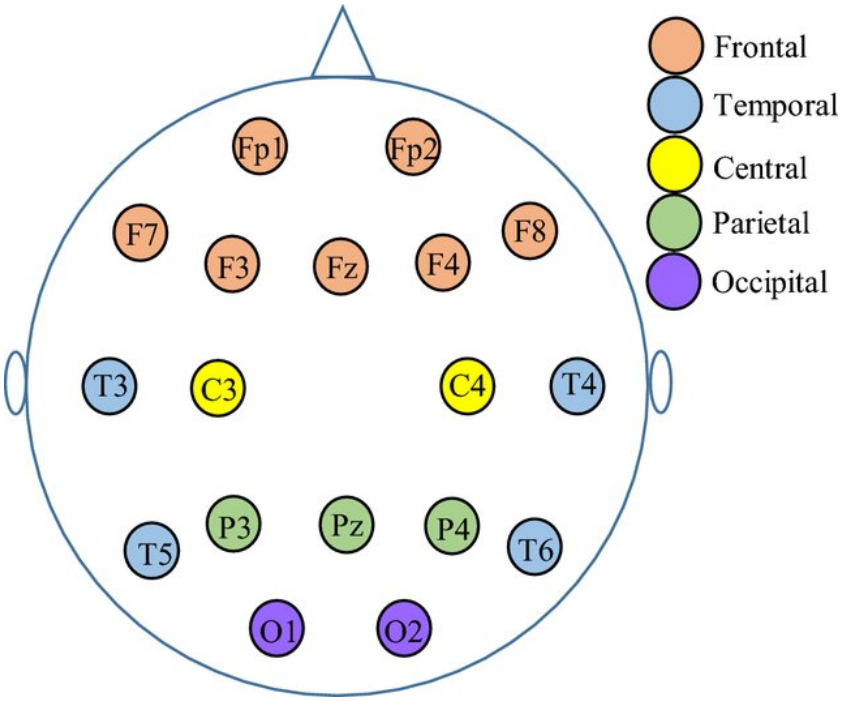


Figure 2.4: EEG electrode placement based on the international 10-20 system. The 18 electrodes cover five major lobes of the brain: Frontal, Temporal, Central, Parietal, and Occipital, adapted from [28].

For this study, a subset of four EEG channels was selected based on their consistent availability across patients and their spatial coverage across the head. Specifically, channels C3, C4, F7, and F8 were chosen. The C3 and C4 electrodes are positioned over the central region of the brain, while F7 and F8 are located over the frontal lobes. Moreover, the odd channels (C3 and F7) cover the left side of the brain, with the even channels (C4 and F8) covering the right side of the brain. In cases where either F7 or F8 was missing (5 and 2 patients respectively), these channels were substituted with more frontal channels Fp1 and Fp2, respectively. These substitute channels were not part of the standard set because they have higher levels of noise due to muscle artifacts. This configuration minimizes the number of channels used, thus data complexity, while preserving coverage of both frontal and central brain areas, as well as both hemispheres, which is important for capturing global patterns in EEG activity. This channel selection also allowed us to use all of the 84 available patients, even though the recordings have different numbers of electrodes due to the circumference of the head of the patients.

2.3.2. Final Input into the Model

After all preprocessing steps, each EEG epoch is formatted as a matrix $\boldsymbol{x}_i$ with dimension $(T, C) = (2000, 4)$. This matrix captures the normalized, resampled and artifact-cleaned signal for a 20 second EEG segment.

For the training of the encoder, the entire dataset of $n = 6,669$ epochs is structured as a 3D tensor with dimensions $(n, T, C) = (6669, 2000, 4)$. This format is compatible with the encoder architecture used for our model. For evaluation and analysis, we maintain a dataframe format of the data. Each row of the dataframe corresponds to one epoch and contains the patient number, epoch number and the patient's outcome label. This structure maintains the hierarchical nature of the data and enables per-patient aggregation for downstream classification tasks.

2.4. Outcome Labels

This study aims to predict long-term neurological outcome following pediatric cardiac arrest. Like in Hunfeld et al. [19], two types of labels are associated with each patient: The clinical outcome at 12 months post arrest, based on the PCPC score, and visual EEG background pattern annotations derived by expert reviewers.

2.4.1. Neurological Outcome: PCPC Score

The primary target variable used is the Pediatric Cerebral Performance Category (PCPC) score at 12 months after cardiac arrest. This scale ranges from 1 (normal neurological function) to 6 (brain death) and provides a standardized measure of global neurological outcome, the definitions of all 6 scores are in Table 2.1. The PCPC is commonly used in pediatric intensive care to quantify neurological outcomes following cardiac arrest [9].

Score	Clinical category
1	Normal at age-appropriate level
2	Mild disability
3	Moderate disability
4	Severe disability
5	Coma or vegetative state
6	Brain death

Table 2.1: Pediatric Cerebral Performance Category (PCPC) scale.

Following Hunfeld et al. [19], the labels were binarized resulting in the two classes:

- **Survival (class 0):** PCPC 1-5,
- **Death (class 1):** PCPC 6.

Let $\boldsymbol{y} = (y_1, \ldots, y_n)$ denote the vector of outcome labels for the n epochs in the dataset, where each label $y_i \in \{0, 1\}$ indicates the binarized outcome class of patient i:

$$y_i = \begin{cases} 0, & \text{if patient } i \text{ has PCPC score in } \{1, 2, 3, 4, 5\}, \\ 1, & \text{if patient } i \text{ has PCPC score } 6. \end{cases} \tag{2.3}$$

Note that all of the epochs belonging to the same patient will have the same outcome label.

In our dataset, no patients are labeled with PCPC score 5 (coma or vegetative state) at 12 months. This reflects the clinical practice in the Netherlands, where continuing life-sustaining treatment in children with no signs of recovery is generally considered as medically futile. In such cases, treatment is typically withdrawn before reaching the 12-month evaluation point. This differs from practices in some other countries, where long-term life support for patients in a persistent vegetative state may be continued indefinitely. Consequently, in our cohort, patients with a very poor prognosis transitioned to PCPC 6 (death) before 12 months, which is captured in the labeling used. The distributions of the original PCPC scores and the resulting binarized labels are shown in Figures 2.5 and 2.6. No patients in the dataset are labeled with PCPC score 5. Out of 84 patients, 47 (56%) are labeled as class 1 (poor outcome), and 37 (44%) as class 0 (favorable outcome).

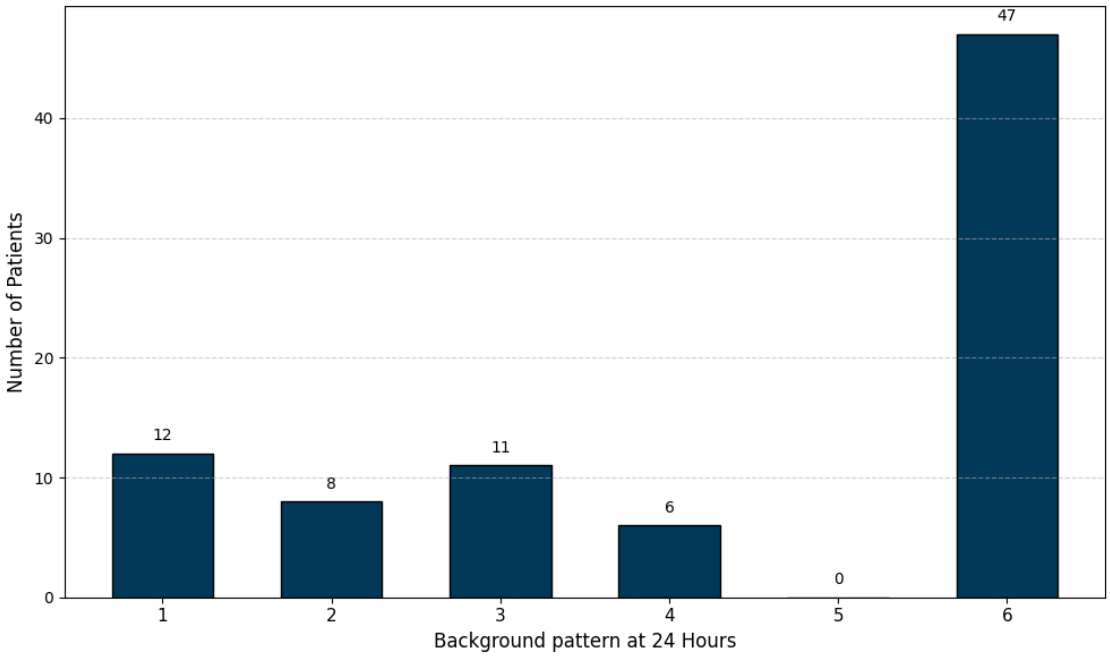


Figure 2.5: Number of patients per PCPC score at 12 months after cardiac arrest.

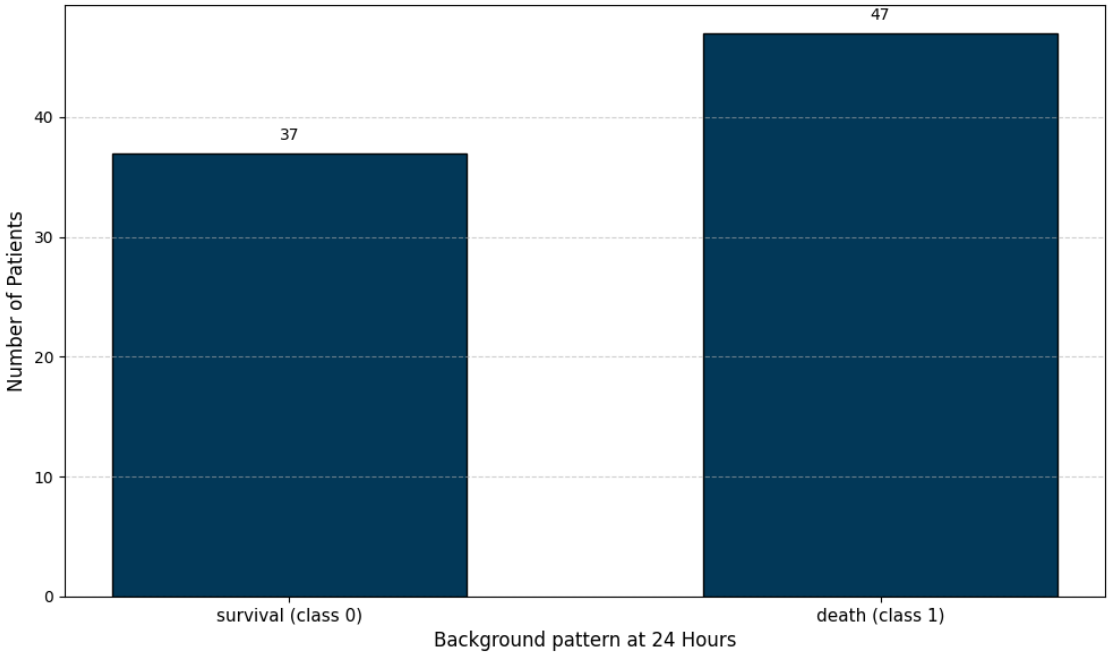


Figure 2.6: Number of patients per binarized PCPC score at 12 months after cardiac arrest.

2.4.2. Visual Analysis of EEG Background Patterns

In addition to PCPC outcome score, each patient in the dataset was also assigned a visual EEG background label, referred to as visual labels in this thesis. These labels reflect qualitative assessments of the EEG background patterns at approximately 24 hours after cardiac arrest, which are also used for our model, and were determined by expert clinical neurophysiologists following standardized protocols [19]. These annotations offer interpretable clinical information about brain function and may help assess whether the encoder captures medically meaningful structure.

The visual classification was performed according to the American Clinical Neurophysiology Society (ACNS) Critical Care EEG terminology [18], as well as the Dutch national protocol for the prognosis of postanoxic encephalopathy in adults. The categories that the EEG background patterns were grouped into are shown in Table 2.2

Score	Clinical category
1	Continuous background pattern with normal amplitudes ($\geq 20\mu V$)
2	Continuous but suppressed background pattern ($< 20\mu V$)
3	No cerebral activity
4	Burst-suppression pattern with identical bursts
5	Burst-suppression pattern with non-identical bursts
6	Generalized periodic discharges on a flat background
7	Generalized periodic discharges on a non-flat background

Table 2.2: Meaning of the visual analysis labels.

Figure 2.7 shows the distribution of these categories in our dataset (al patients with score 0 have already been removed from the dataset during preprocessing). The most frequent label is 1 (”continuous background pattern with normal amplitudes ($\geq 20\mu V$)”), which is associated with a substantially higher survival rate at 12 months. Notably, none of the patients with a background pattern other than continuous with amplitudes $\geq 20\mu V$ survived beyond 12 months, this is shown in Figure 2.8

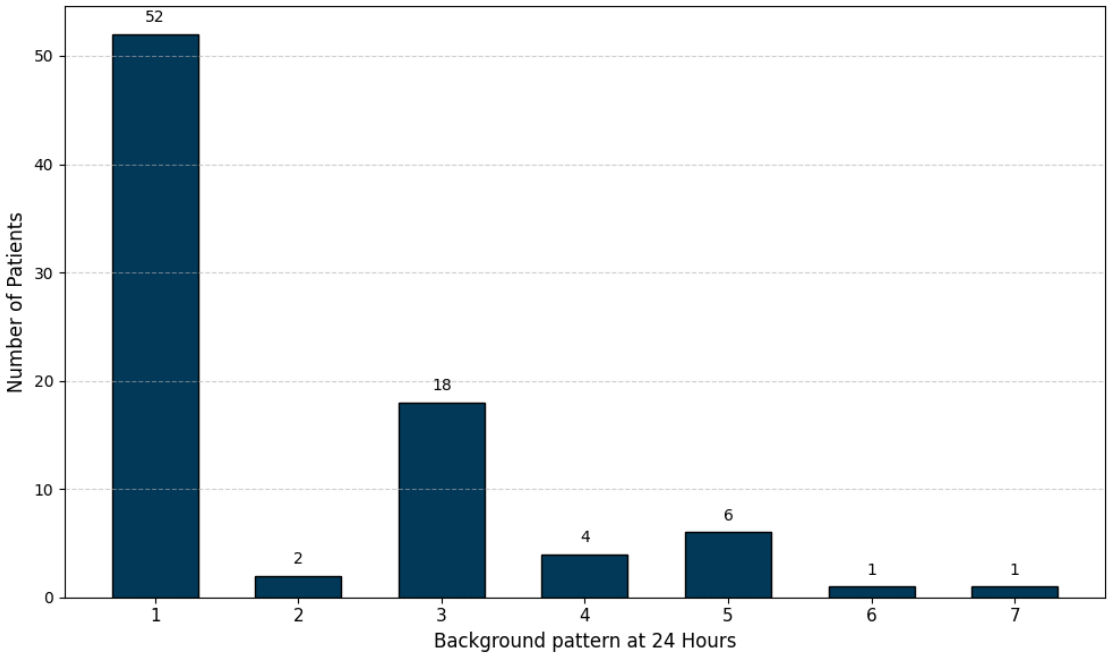


Figure 2.7: Distribution of visual EEG background labels in the dataset. These labels are derived from 30-minute EEG segments recorded approximately 24 hours after return of spontaneous circulation.

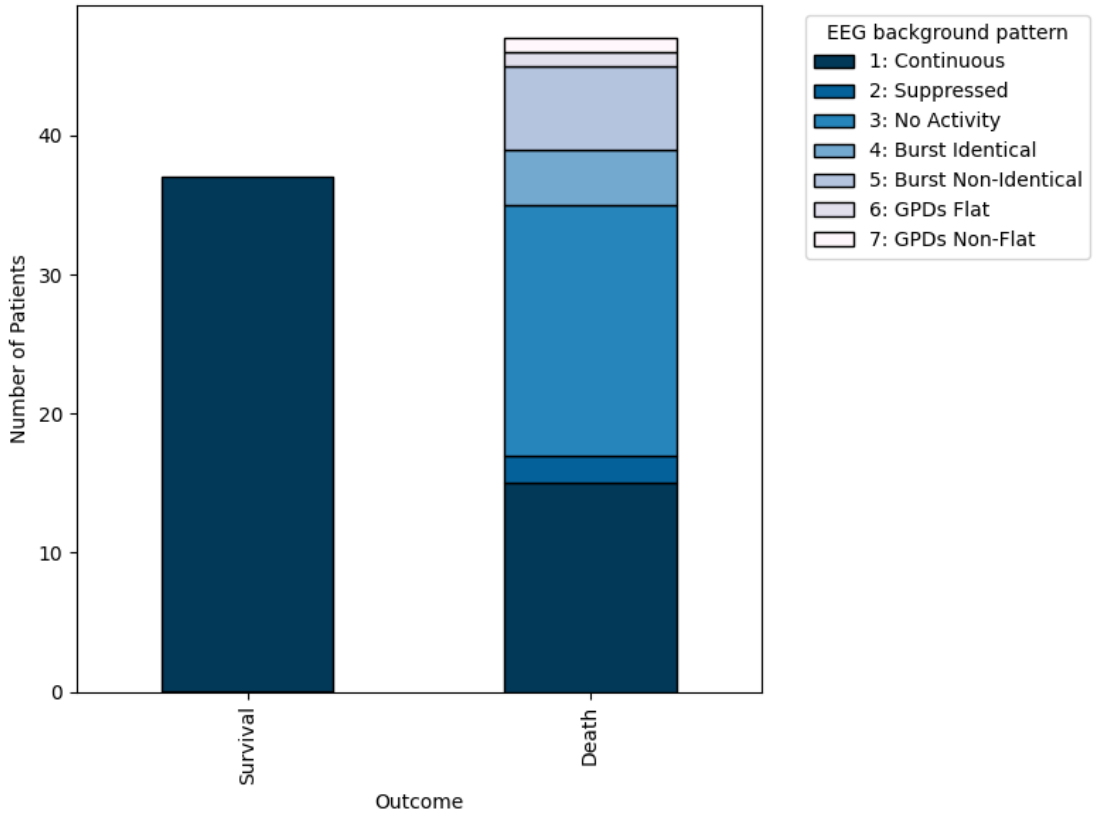


Figure 2.8: The survival status of children 12 months after resuscitation. The colors indicate the background pattern of EEG registered at 24 hours after cardiac arrest.

While these labels are not used as targets during training, we use them in auxiliary analyses (see Section 5.1) to evaluate whether the encoder representations align with clinical expert judgments. Accurate clustering by visual label would suggest that the model captures physiologically meaningful features, even without supervision.

3

Contrastive Learning

In this thesis, we first train a TS2Vec encoder, which uses contrastive learning with a convolutional neural network (CNN), on unlabeled EEG data to extract meaningful representations. These representations are then used as input features for a supervised classification model. This section introduces the core techniques used for the encoder.

3.1. Introduction to Contrastive Learning

Machine learning is a field of artificial intelligence that enables computers to learn patterns from data. These learned patterns can consequently be used to make predictions or decisions without these prediction or decision rules having to be explicitly programmed. Traditionally, machine learning has two different approaches to learning from data: supervised learning and unsupervised learning [42].

Supervised learning relies on labeled data, which means that every training sample $\boldsymbol{x}_i \in \mathbb{R}^d$ has a corresponding target output or label $y_i \in \mathcal{Y}$. The model learns a function $f(\cdot)$ that minimizes the difference between the predicted label $\hat{y}_i = f(\boldsymbol{x}_i)$ and the true label y_i. This approach is widely used in regression [7][33] and classification tasks (like predicting whether a patient will recover based on qEEG signals) [6][24].

Unsupervised learning, in contrast, deals with unlabeled data. Instead of learning from predefined targets, the model attempts to identify structure within the dataset, such as patterns, clusters, or low-dimensional manifolds, without external supervision. These discovered patterns are used to build internal representations of the data, which aim to capture essential information about its underlying structure. Such representations can serve as inputs for downstream tasks, such as classification, clustering, or anomaly detection. Common unsupervised techniques include dimensionality reduction [38] and clustering (e.g., grouping patients with similar EEG patterns) [20][11].

Supervised learning has traditionally been the dominant approach in machine learning due to its ability to exploit the information given by the labels [42]. However, acquiring such labels can be costly and time-consuming, especially in fields like healthcare where expert annotation is required. Moreover, in some applications, it is fundamentally difficult to define the correct label for the data. EEG analysis provides a good example. Although outcome labels are known for each patient, it is unclear how individual EEG segments should be labeled, as they may contain a mix of pathological and non-pathological activity. This uncertainty reduces the efficiency of supervised learning, since the training labels may not perfectly correspond to the underlying signal characteristics. Nevertheless, the availability of patient-level outcome labels still allows supervised methods to achieve significantly better performance than purely unsupervised approaches, as even imperfect supervision can steer the model toward clinically meaningful patterns. In contrast, unsupervised learning must infer structure based only on internal relationships within the data, without any external guidance, and often produces representations that are less optimized for specific downstream tasks, typically leading to a lower performance[12].

Self-supervised learning has emerged as an alternative that bridges the gap between supervised and

unsupervised learning [43][10]. It enables models to learn useful feature representations from unlabeled data by defining specific learning objectives known as pretext tasks. These tasks allow the model to generate its own supervision signal by leveraging structural properties of the data. A simple example is in in image data: we give a model 2 views of the same image, one is the original and one is rotated by a certain angle. The pretext task would be asking the model to identify what the rotation angle is. Solving this task forces the model to figure out the content of the image without needing a label of the content of the image [15]. The rotation identification task is not the main interest, but it serves to induce meaningful representations (one could say it serves to give the model a better understanding of the contents of the image). Therefore it is considered a pretext task, it prepares the model for more relevant downstream tasks.

This thesis applies a self-supervised learning technique called contrastive learning [8] [22]. Rather than learning from prespecified labels, contrastive learning trains a model to pull semantically similar samples (positive pairs) closer together in a latent space, while pushing dissimilar samples (negative pairs) farther apart. The latent space is a lower-dimensional vector space in which the most informative aspects of the input data are encoded. The objective of distinguishing between such sample pairs constitutes a pretext task, which serves to structure the learned representations in the absence of explicit labels, or supervision. Formally, the model learns a representation function $f(\cdot)$ that maps an input $\boldsymbol{x}_i$ to a vector $\boldsymbol{r}_i = f(\boldsymbol{x}_i) \in \mathbb{R}^R$, such that semantic similarity in the input domain corresponds to geometric proximity in the latent space.

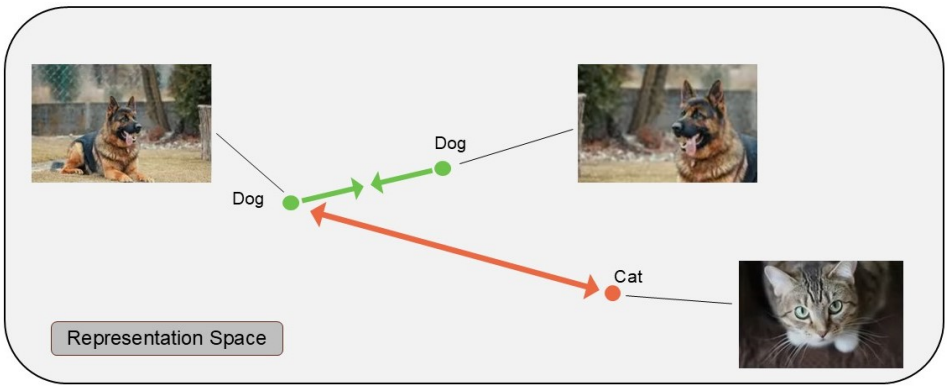


Figure 3.1: A visualization of contrastive loss in a representation space. The two views of the same image of the dog are pulled together, while the image of dog and the cat are pushed farther apart.

A common approach to contrastive learning in computer vision involves generating positive pairs through carefully designed data augmentations. For instance, one can take a single image and produce a modified version by applying transformations such as random cropping, flipping, rotation, or masking some parts of the image. These augmentations are constructed to preserve the identity of the image (e.g., a rotated picture of a dog is still semantically a dog) while altering its appearance enough to make the task non-trivial. Each positive pair consists of the original and an augmented image, encouraging the model to learn invariant features. Negative samples are drawn from other images in the dataset and are assumed to belong to different semantic categories. By contrasting these pairs, the model learns to extract consistent, high-level features that generalize well across tasks. Figure 3.1 illustrates this idea, showing how the model organizes latent representations by pulling similar images (e.g., two images of the same dog) together and pushing dissimilar ones (e.g., an image of a dog vs an image of a cat) apart.

By learning to distinguish between positive and negative pairs, the model is encouraged to identify stable, consistent patterns in the data that define similarity. This learning objective can be optimized without using any labels. Contrastive learning has proven particularly effective in domains like computer vision [10], natural language processing [51], and, more recently, time series analysis [31]. Its ability to extract informative representations from unlabeled data makes it especially useful in medical applications, where labeled data is expensive or scarce.

3.2. Contrastive Learning for Time Series

Training a contrastive learning model for time series typically follows a process consisting of two main steps. First, an encoder network transforms the time series input into a low-dimensional latent space representation. Then, a contrastive loss function is used to optimize the encoder such that the representations of similar inputs are close together in the latent space, while dissimilar inputs are mapped far apart. This training process is illustrated in the top panel of Figure 3.2.

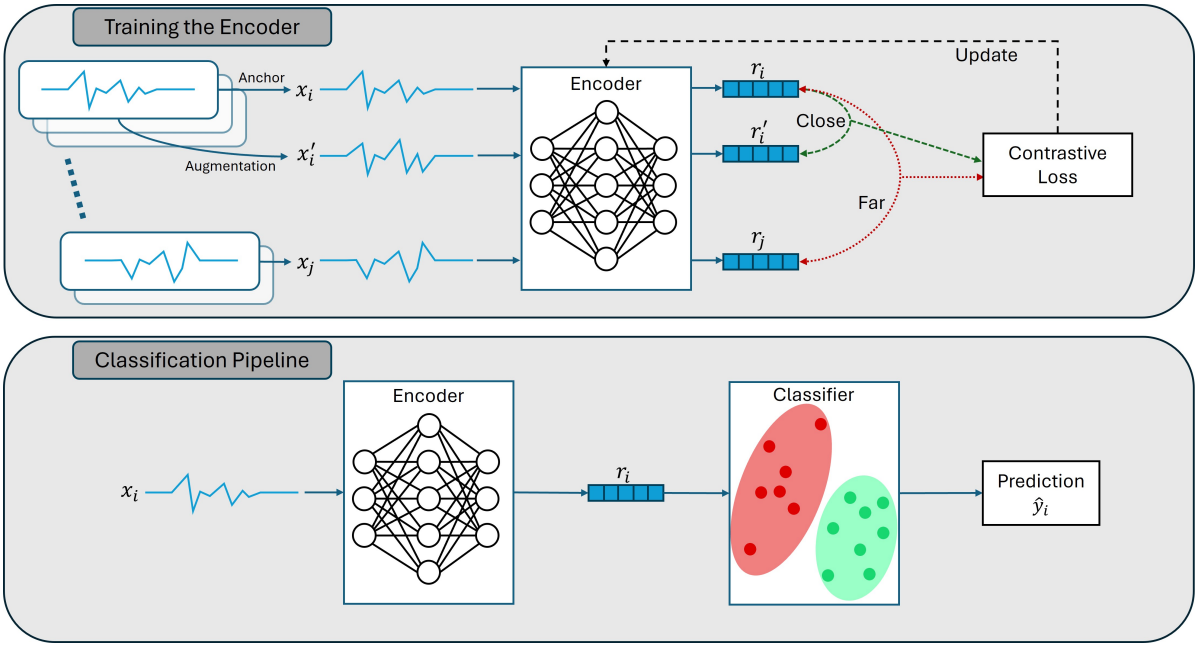


Figure 3.2: The general architecture of a contrastive learning model. The top panel shows the training process of the encoder (from left to right), where an original sample (anchor), an augmented version, and a negative sample are encoded and used to compute a contrastive loss. The bottom panel shows the classification pipeline after training, where the encoder and classifier are used together to make predictions.

The encoder is a neural network, denoted by $f(\cdot;\Theta)$ with parameters Θ, which maps an input sample $\boldsymbol{x}_i$ to a vector representation $\boldsymbol{r}_i$ in a latent representation space $\mathbb{R}^R$:

$$\boldsymbol{r}_i = f(\boldsymbol{x}_i;\Theta). \tag{3.1}$$

Common architecture choices for f include convolutional or recurrent neural networks, which are well-suited to modeling local and sequential structure in time series data.

A key design choice in contrastive learning is how to define a positive pair. In our setting, a positive pair $(\boldsymbol{x}_i, \boldsymbol{x}_i')$ is formed by taking a single input $\boldsymbol{x}_i$ and applying controlled transformations to obtain an augmented version $\boldsymbol{x}_i'$. The original input is often called the anchor. All other samples $\boldsymbol{x}_j$ in the dataset, including their augmentations $\boldsymbol{x}_j'$, are treated as negative samples with respect to $\boldsymbol{x}_i$.

For image data, positive pairs are typically created using augmentations such as random cropping, flipping, or rotation. For time series, however, the choice of augmentation is more delicate: transformations must preserve the semantic content of the signal without distorting features relevant to the downstream task. In this thesis, we follow the TS2Vec framework and use two augmentations: random cropping and timestamp masking. Random cropping selects a fixed-length subsegment of the original time series, encouraging the model to learn representations that are robust across local windows. Timestamp masking involves suppressing parts of the input from the encoder, forcing the encoder to rely on surrounding context to infer informative features. Both augmentations are designed to preserve the physiological identity of the EEG while introducing variability in presentation. They are described in more detail in Section 3.3.3. Other augmentations common in time series contrastive learning, such as time warping, are avoided here, as they may distort clinically meaningful

EEG characteristics.

In practice, computing the contrastive loss over all possible negative samples in the dataset is computationally infeasible. It would require storing all latent representations in memory and comparing each anchor to every other sample, which is prohibitively expensive for large datasets. To address this, contrastive learning methods typically approximate the full negative set using only a current mini-batch.

In our implementation, a batch of 8 original EEG samples is drawn from the training set at each iteration. For each of these inputs we generate a single augmented version resulting in 16 inputs per training batch: 8 anchors and 8 augmentations. These are then passed through the encoder to obtain 16 latent representations. So the batch $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_8$ becomes training batch $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_8, \boldsymbol{r}'_1, \ldots, \boldsymbol{r}'_8$. Each representation $\boldsymbol{r}_i$ (or $\boldsymbol{r}'_i$) has one designated positive sample and 14 negatives within the batch.

This augmentation-based expansion allows the model to learn from pairwise comparisons while keeping memory requirements low. Given there are, on average, $\frac{4}{5} \cdot 6669 \approx 53335$ training segments and a batch size $B = 8$ (referring to original inputs), each training epoch consists of approximately $\frac{n}{B} = \frac{5335}{8} \approx 667$ iterations (rounded up). The full training process is repeated for six epochs to allow the encoder to refine its representations across multiple passes through the data.

Once positive and negative pairs have been defined, a contrastive loss function is applied to evaluate how well the encoder separates similar from dissimilar inputs in latent space. The general contrastive loss function (3.2) encourages the model to maximize the similarity between positive pairs while minimizing similarity to negative samples. Let $\boldsymbol{r}_i = f(\boldsymbol{x}_i; \Theta)$ and $\boldsymbol{r}'_i = f(\boldsymbol{x}'_i; \Theta)$ be the latent representations of a positive pair $(\boldsymbol{x}_i, \boldsymbol{x}'_i)$. Let $\{\boldsymbol{r}_j\}_{j=1}^B \cup \{\boldsymbol{r}'_j\}_{j=1}^B$ denote the representations of all samples in the training batch. Then the general contrastive loss is defined as:

$$\mathcal{L}_{i,j}^{\text{contrastive}} = -\log \frac{\exp(\mathsf{sim}(\boldsymbol{r}_i, \boldsymbol{r}'_i))}{\sum_{j=1}^{B} \left[\exp(\mathsf{sim}(\boldsymbol{r}_i, \boldsymbol{r}'_j)) + 1_{[j \neq i]} \exp(\mathsf{sim}(\boldsymbol{r}_i, \boldsymbol{r}_j))\right]}, \tag{3.2}$$

where $\mathsf{sim}(\cdot, \cdot)$ is a similarity function, like the cosine similarity:

$$\mathsf{sim}(\boldsymbol{r}_i, \boldsymbol{r}_j) = \frac{\boldsymbol{r}_i^\top \boldsymbol{r}_j}{\|\boldsymbol{r}_i\| \cdot \|\boldsymbol{r}_j\|}, \tag{3.3}$$

or the inner product (dot product) of two vectors $\mathsf{sim}(\boldsymbol{r}_i, \boldsymbol{r}_j) = \boldsymbol{r}_i^\top \boldsymbol{r}_j$.

After training, the contrastive loss is no longer used. The encoder $f(\cdot; \Theta)$ has learned to map similar EEG segments close together in latent space and dissimilar ones farther apart. It can then be used independently to extract representations $\boldsymbol{r}_i$ from new, unseen inputs $\boldsymbol{x}_i$. These representations can serve as input features for downstream tasks such as clustering, anomaly detection, or classification—the focus of this thesis.

The encoder used in this project follows the TS2Vec model [53], which builds on the general contrastive loss and incorporates design decisions specifically tailored to the temporal and hierarchical structure of time series data.

3.3. Our implementation: Contrastive Learning on EEG data

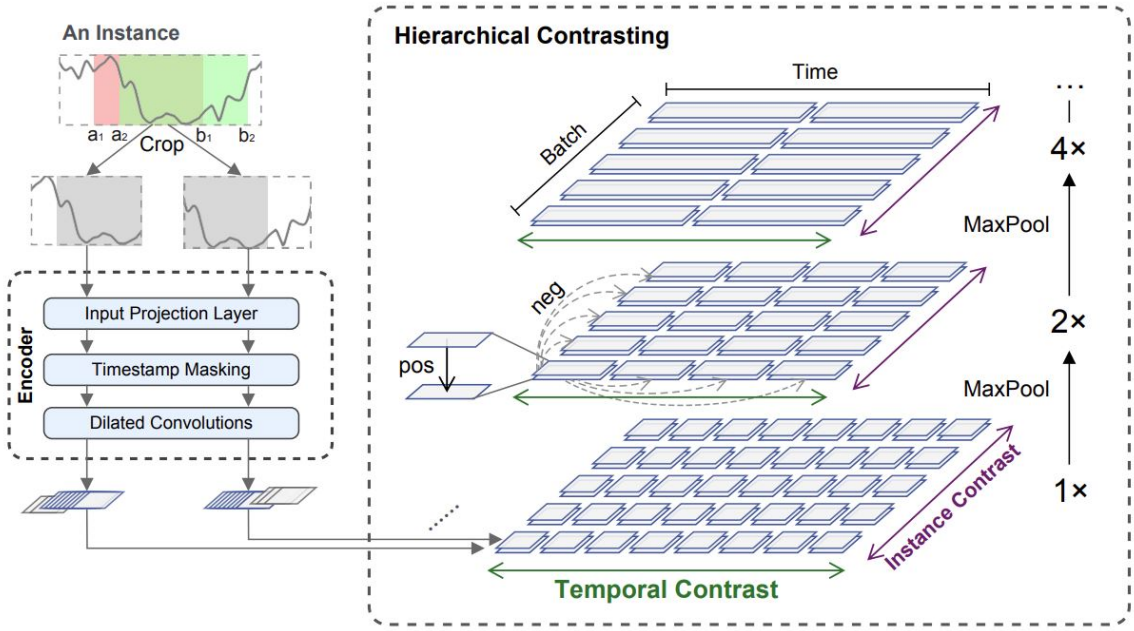


Figure 3.3: The architecture of TS2Vec. Although this figure shows a univariate time series as the input example, the framework supports multivariate input. Each parallelogram denotes the representation vector on a timestamp of an instance. (Source: TS2Vec, Yue et al. [53])

To learn meaningful representations from raw EEG data without supervision, we implement TS2Vec, short for Time Series to Vector, a self-supervised contrastive learning framework specifically designed for time series [53]. We selected TS2Vec because it is a highly versatile method that has demonstrated strong performance across a wide range of time series tasks and domains. In its 2022 publication, TS2Vec outperforms previous state-of-the-art methods on 125 univariate (UCR) and 29 multivariate (UEA) benchmark datasets, including various physiological signals such as electrocardiogram (ECG) and EEG signals. Furthermore, a systematic review by Zhang et al. [54] in 2023 found that TS2Vec remained one of the strongest contrastive learning methods for time series, showing competitive or superior performance across classification, forecasting, and anomaly detection tasks. Its performance was particularly strong on medical time series classification problems. This robustness and generalization capability (benchmarked on over 150 datasets) make TS2Vec particularly well-suited for clinical applications, where data characteristics and label availability can vary significantly. In addition, its encoder architecture is flexible and modular, supporting a decoupled encoder–classifier design that suits the requirements of this novel pediatric EEG setting.

The encoder is trained on unlabeled EEG epochs and serves as the feature extractor for downstream classification tasks (see Section 4). For every cross-validation split, the TS2Vec encoder was trained exclusively on the unlabeled epochs from the training patients; no data from the test patients was seen during this phase. This section describes the model architecture, the training setup and the augmentations used, the architecture of the model is shown in Figure 3.3.

3.3.1. Convolutional Encoder Architecture

The encoder architecture follows the exact design specified in TS2Vec [53]. It transforms a multichannel EEG segment into a structured latent representation using a deep convolutional network with four main components: cropping of the input signal to create two augmented views of the input (x_i and x_i', see Section 3.3.3), an input projection layer, a timestamp masking module, and a stack of ten residual dilated convolutional blocks (see Appendix A).

Each 20-second EEG segment is represented as a multivariate time series $\boldsymbol{x}_i \in \mathbb{R}^{T \times C}$, where T and C denote the number of time points and channels respectively. After cropping the signal into two augmented views, the input projection layer maps each C-dimensional observation at time t, denoted $\boldsymbol{x}_{i,t} \in \mathbb{R}^C$, to a 64-dimensional vector independently across time:

$$\boldsymbol{u}_{i,t} = \boldsymbol{W}_{\text{in}} \boldsymbol{x}_{i,t} + \boldsymbol{b}_{\text{in}} \quad \in \mathbb{R}^{64}, \tag{3.4}$$

where $\boldsymbol{W}_{\text{in}} \in \mathbb{R}^{64 \times 4}$ and $\boldsymbol{b}_{\text{in}} \in \mathbb{R}^{64}$ are learnable parameters, producing a sequence $\boldsymbol{U}_i = (\boldsymbol{u}_{i,1}, \ldots, \boldsymbol{u}_{i,T}) \in \mathbb{R}^{T \times 64}$. This step increases the expressiveness of the signal by embedding it into a higher-dimensional latent space. The model is then able to represent more complex patterns and interactions between channels before any temporal feature extraction begins. This is analogous to expanding the feature space in classical machine learning to allow for more flexible decision boundaries.

After the input projection layer, timestamp masking is applied on $\boldsymbol{U}_i$. Here a randomly selected subset of timestamps is replaced with zero vectors to promote the encoder to learn more robust representations. As this is a way of creating positive samples, the exact method is explained in Section 3.3.3.

The core of the encoder is a diluted CNN consisting of ten residual blocks, each with two 1D convolutional layers followed by GELU activations. These layers extract temporal patterns across multiple scales. All convolutions use kernel size $K = 3$ and symmetric zero-padding to preserve sequence length. Dilation is used to increase the receptive field exponentially across depth, allowing each output vector to attend to a large temporal context. Specifically, block ℓ uses dilation $d_\ell = 2^{\ell-1}$. The final block increases the feature dimension from 64 to 320 to match the desired representation output size. A summary of the architecture is provided in Table 3.1. The mathematical description of the dilated CNN can be found in Appendix A.

Block ℓ	Dilation d_ℓ	In dim. $D^{\ell-1}$	Out dim. D^ℓ	Convolutional layers	Residual projection
1–9	$2^{\ell-1}$	64	64	2	no
10	512	64	320	2	1×1 conv

Table 3.1: Summary of the dilated convolutional encoder. All convolutions use kernel size $K = 3$, with zero padding on both sides to preserve input length.

The output of the tenth, and final, convolutional block is the latent representation sequence $\boldsymbol{H}_i = f(\boldsymbol{x}_i; \Theta) \in \mathbb{R}^{T \times R}$, where each row $\boldsymbol{h}_{i,t} \in \mathbb{R}^R$ corresponds to the representation at timestep t for input segment $\boldsymbol{x}_i$. This sequence forms the basis for the contrastive learning objective described in the next section. The training process optimizes the encoder parameters Θ so that $\boldsymbol{H}_i$ captures stable and informative patterns from raw EEG data, even in the absence of labels.

To obtain a fixed-length representation for downstream classification, a temporal max pooling operation is applied across the entire time axis. Formally, the pooled representation $\boldsymbol{r}_i \in \mathbb{R}^{320}$ is defined as:

$$\boldsymbol{r}_i[k] = \max_{1 \le t \le T} \boldsymbol{h}_{i,t}[k], \quad \text{for } k = 1, \ldots, 320, \tag{3.5}$$

where $\boldsymbol{r}_i[k]$ denotes the k-th feature of the vector $\boldsymbol{r}_i$ and $\boldsymbol{h}_{i,t}[k]$ denotes the k-th feature of the vector $\boldsymbol{h}_{i,t}$. That is, max pooling selects the maximum value across all time steps for each feature channel independently. Max pooling is commonly used in contrastive learning for time series, as it preserves the most salient temporal features while discarding less informative fluctuations. It enables efficient and consistent input to downstream models and avoids the need for task-specific temporal aggregation strategies. The resulting pooled vector $\boldsymbol{r}_i \in \mathbb{R}^R$ serves as a compact summary of the EEG epoch and is used as input to the classifier.

3.3.2. Contrastive Learning Objective: the Loss Function

Once the encoder has produced a latent representation $\boldsymbol{H}_i = f(\boldsymbol{x}_i; \Theta) \in \mathbb{R}^{T \times R}$ for a time series input $\boldsymbol{x}_i \in \mathbb{R}^{T \times C}$, the network is trained using a contrastive loss function. The goal of this loss is to

encourage the encoder to produce similar outputs for semantically similar inputs, and dissimilar outputs for unrelated inputs, without requiring any labels.

To achieve this, TS2Vec defines two types of contrastive objectives: a temporal contrastive loss and an instance-wise contrastive loss. These two objectives complement each other: temporal loss encourages discrimination across different timesteps, instance loss encourages discrimination across different samples. Both losses are computed per timestep, for each sample in the batch.

Let $B = 8$ denote the batch size, which is the number of samples processed each batch, and T the number of timesteps. For each sample $\boldsymbol{x}_i$, we generate two augmented versions and compute their encoded sequences:

$$\boldsymbol{H}_i = (\boldsymbol{h}_{i,1}, \ldots, \boldsymbol{h}_{i,T}), \quad \boldsymbol{H}'_i = (\boldsymbol{h}'_{i,1}, \ldots, \boldsymbol{h}'_{i,T}), \quad \text{with } \boldsymbol{h}_{i,t}, \boldsymbol{h}'_{i,t} \in \mathbb{R}^R.$$

The temporal contrastive loss operates across timesteps. For each time step t, it encourages agreement between $\boldsymbol{h}_{i,t}$ and $\boldsymbol{h}'_{i,t}$ from the same sample i, while discouraging similarity with other timesteps of the two samples. It is defined as:

$$\mathcal{L}_{i,t}^{\text{temp}} = -\log \frac{\exp\left(\boldsymbol{h}_{i,t}^\top \boldsymbol{h}'_{i,t}\right)}{\sum_{t'=1}^{T}\left[\exp\left(\boldsymbol{h}_{i,t}^\top \boldsymbol{h}'_{i,t'}\right) + 1_{[t\neq t']}\exp\left(\boldsymbol{h}_{i,t}^\top \boldsymbol{h}_{i,t'}\right)\right]} \tag{3.6}$$

Here, the numerator is the similarity between $\boldsymbol{h}_{i,t}$ and its positive counterpart $\boldsymbol{h}'_{i,t}$, while the denominator includes negative examples from other timesteps in both views. The dot product serves as the similarity metric, and $1_{[t\neq t']}$ is an indicator function to avoid self-comparisons in the denominator.

The instance-wise contrastive loss operates across samples. For each time step t, it encourages agreement between $\boldsymbol{h}_{i,t}$ and $\boldsymbol{h}'_{i,t}$ from the same sample i, while discouraging similarity to representations of other samples, $j \neq i$, at the same timestep:

$$\mathcal{L}_{i,t}^{\text{inst}} = -\log \frac{\exp\left(\boldsymbol{h}_{i,t}^\top \boldsymbol{h}'_{i,t}\right)}{\sum_{j=1}^{B}\left[\exp\left(\boldsymbol{h}_{i,t}^\top \boldsymbol{h}'_{j,t}\right) + 1_{[i\neq j]}\exp\left(\boldsymbol{h}_{i,t}^\top \boldsymbol{h}_{j,t}\right)\right]} \tag{3.7}$$

The total contrastive loss at the original resolution is computed by averaging both losses over the batch and all timesteps:

$$\mathcal{L}_{\text{dual}} = \frac{1}{BT}\sum_{i=1}^{B}\sum_{t=1}^{T}\left(\mathcal{L}_{i,t}^{\text{temp}} + \mathcal{L}_{i,t}^{\text{inst}}\right) \tag{3.8}$$

To capture temporal structure at multiple scales, TS2Vec applies the contrastive loss hierarchically. This is done by computing the dual contrastive loss $\mathcal{L}_{\text{dual}}$ not only on the original representation sequence $\boldsymbol{H}_i$, but also on increasingly coarse versions of it. After each loss computation, the representation sequence is downsampled using 1D max pooling with kernel size 2 along the temporal axis, effectively halving the number of timesteps by taking the max value of every two timesteps. This yields a new, lower-resolution sequence on which the loss is computed again. The process is repeated recursively until the sequence is reduced to a single timestep. This process is visualized in Figure 3.3. Let d denote the total number of levels in the resulting hierarchy, so the number of times that the 1D max pooling can be applied. The final hierarchical contrastive loss is defined as the average over the losses at each resolution:

$$\mathcal{L}_{\text{hier}} = \frac{1}{d}\sum_{s=1}^{d}\mathcal{L}_{\text{dual}}^{(s)}, \tag{3.9}$$

where $\mathcal{L}_{\text{dual}}^{(s)}$ denotes the dual loss at resolution level s. The algorithm for the hierarchical loss is given in Algorithm 1. This multiscale training objective encourages the encoder to learn representations that are stable and informative across different temporal resolutions, which is particularly appropriate for EEG signals exhibiting structure at both short and long timescales.

Algorithm 1 Calculating the hierarchical contrastive loss

procedure HIERLOSS(r, r')
  $\mathcal{L}_{\text{hier}} \leftarrow \mathcal{L}_{\text{dual}}(r, r')$
  $d \leftarrow 1$
  while time_length(r) > 1 **do**
    // The maxpool1d operates along the time axis.
    $r \leftarrow \text{maxpool1d}(r, \text{kernel\_size} = 2)$
    $r' \leftarrow \text{maxpool1d}(r', \text{kernel\_size} = 2)$
    $\mathcal{L}_{\text{hier}} \leftarrow \mathcal{L}_{\text{hier}} + \mathcal{L}_{\text{dual}}(r, r')$
    $d \leftarrow d + 1$
  end while
  $\mathcal{L}_{\text{hier}} \leftarrow \mathcal{L}_{\text{hier}}/d$
  return $\mathcal{L}_{\text{hier}}$
end procedure

3.3.3. Positive Sample Pairs

A key design choice of any contrastive learner is how we construct positive sample pairs, two different views of the same EEG input. These views are created through controlled augmentations that alter the input while preserving the semantic information in the input. The encoder is then trained to produce similar latent representations for the two views, learning robust and invariant features. In our implementation, the two augmentation strategies from the TS2Vec framework are: random cropping and timestamp masking, both of which are explained below. These augmentations were chosen to preserve the clinical meaning of the EEG signal while helping the encoder to learn features that are stable across small changes in the input. The augmentations are only applied to the input data in the training phase.

EEG recordings from comatose patients tend to show reduced variability over time, because of the diminished responsiveness of the brain and the effects of hypoxic injury following resuscitation. As a result, different but overlapping segments of the same EEG are likely to contain similar information. Random cropping leverages this by forcing the encoder to recognize consistent patterns, regardless of their absolute position within the time window. Timestamp masking serves a complementary computational purpose, it prevents the encoder from over-relying on individual timestamps by artificially removing values. This promotes more distributed and context-aware representations. Together, the augmentations and the contrastive loss define a contrastive self-supervised pretext task that trains the encoder to extract stable and clinically meaningful features from EEG recordings.

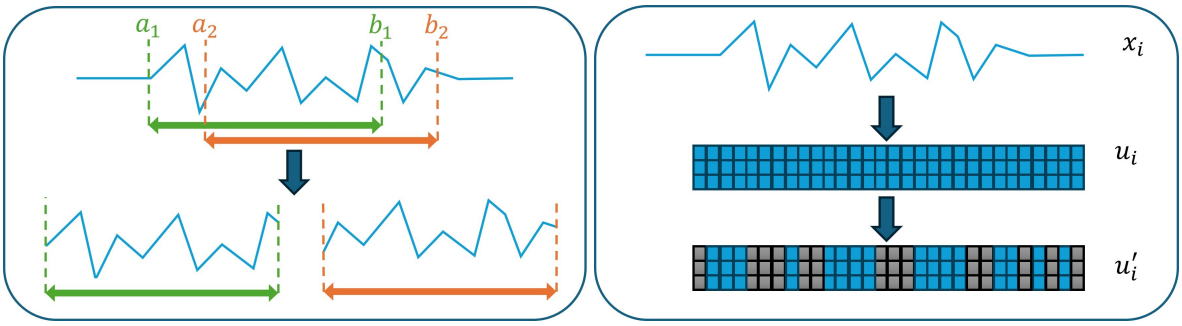
Random cropping. Let $\boldsymbol{x}_i \in \mathbb{R}^{T \times C}$ denote a multivariate EEG time series with T timesteps and C channels. To create a positive pair, we randomly select two overlapping subsegments, defined by intervals $[a_1, b_1]$ and $[a_2, b_2]$, such that $0 < a_1 \leq a_2 \leq b_1 \leq b_2 \leq T$ and the overlap $[a_2, b_1]$ is non-empty. These define two views of the same EEG input: $\boldsymbol{x}_i^{(1)} = \boldsymbol{x}_i[a_1 : b_1]$ and $\boldsymbol{x}_i^{(2)} = \boldsymbol{x}_i[a_2 : b_2]$. For consistency, we will denote these as $\boldsymbol{x}_i$ and $\boldsymbol{x}_i'$. Figure 3.4a shows an example of random cropping on a time series. This augmentation encourages the encoder to learn features invariant to absolute temporal position in the given input, this is a clinically appropriate assumption in EEG from comatose patients, where relevant features tend to persist over time.

Timestamp masking. To further promote robustness of the encoder, we apply timestamp masking independently to each view after the input projection layer and before the convolutional encoder, denoted as $\boldsymbol{U}_i = (\boldsymbol{u}_{i,1}, \ldots, \boldsymbol{u}_{i,T}) \in \mathbb{R}^{T \times 64}$ as shown in Figure 3.4b. Specifically, it masks the latent sequence along the temporal axis using a binary mask $\boldsymbol{m}_i \in \{0, 1\}^T$ sampled from a Bernoulli distribution with $p = 0.5$. The masked representation is then obtained by element-wise multiplication:

$$\tilde{\boldsymbol{u}}_{i,t} = m_t \cdot \boldsymbol{u}_{i,t} \text{ for } t = 1, \ldots, T \tag{3.10}$$

This operation is applied stochastically at each training step. while not physiologically motivated, this augmentation serves to prevent the encoder from overfitting to specific timepoints and instead

encourages reliance on distributed temporal context. In effect, it regularized the model by forcing it to reconstruct stable representations even when parts of the input are hidden. This improves generalization under signal variability, such as brief artifacts or missing data.



(a) Random cropping **(b)** Timestamp masking.

Figure 3.4: The augmentation methods used in the encoder.

Together, these two augmentations define the transformation space over which the encoder must learn to remain invariant. By presenting partially overlapping and partially obscured views of the same input, the model is trained to extract robust and generalizable features. These augmentations, in combination with the contrastive loss, for the foundation of the self-supervised pretext task used to train the encoder.

3.4. Evaluation Methods for the Encoder

After training the encoder $f(\cdot;\Theta)$ using the contrastive learning objectives described above, we seek to understand the structure and content of the learned representations $\boldsymbol{r}_i \in \mathbb{R}^R$. Gaining insight into these representations allows us to investigate whether the model is capturing structured, meaningful features that are suitable for downstream tasks. Unlike supervised models, which can be evaluated directly against a ground truth label, the performance of an unsupervised encoder is more difficult to assess. There is no intrinsic measure of representation quality without linking to a downstream task [26].

This difficulty reflects a broader challenge in neural network research. Despite the widespread success of deep learning models, there is often limited interpretability or theoretical understanding of what internal representations capture, or how meaningful they are for downstream use cases [47]. This is sometimes referred to as the 'black box' nature of deep models. For contrastive learning in particular, evaluating the encoder in isolation, without reference to a classifier or prediction task, offers limited quantitative feedback on whether the learned representations are useful.

As a result, the evaluation of the encoder in this project is necessarily qualitative. We apply one main technique to explore the structure of the learned latent space: t-SNE is used to visualize whether the encoder groups similar inputs together in a way that reflects patient identity or clinical outcome. In addition, we include saliency maps in Section 3.4.2 as an exploratory tool to investigate which parts of the EEG input influence the encoder's representation. However, these maps are difficult to interpret reliably and are not used to support any formal conclusions.

Although these techniques do not quantify performance in a strict sense, they provide intuition about the encoder's behavior and offer limited insight into whether it has learned clinically meaningful structure.

3.4.1. t-SNE: Visualization of Encoder Representations

The t-distributed Stochastic Neighbor Embedding (t-SNE) algorithm is a nonlinear dimensionality reduction technique used to visualize high-dimensional data in a lower-dimensional (typically 2D) space [30]. It aims to preserve local structure by mapping nearby points in the original space to nearby points in the projected space. Figure 3.5 shows an example of a t-SNE plot.

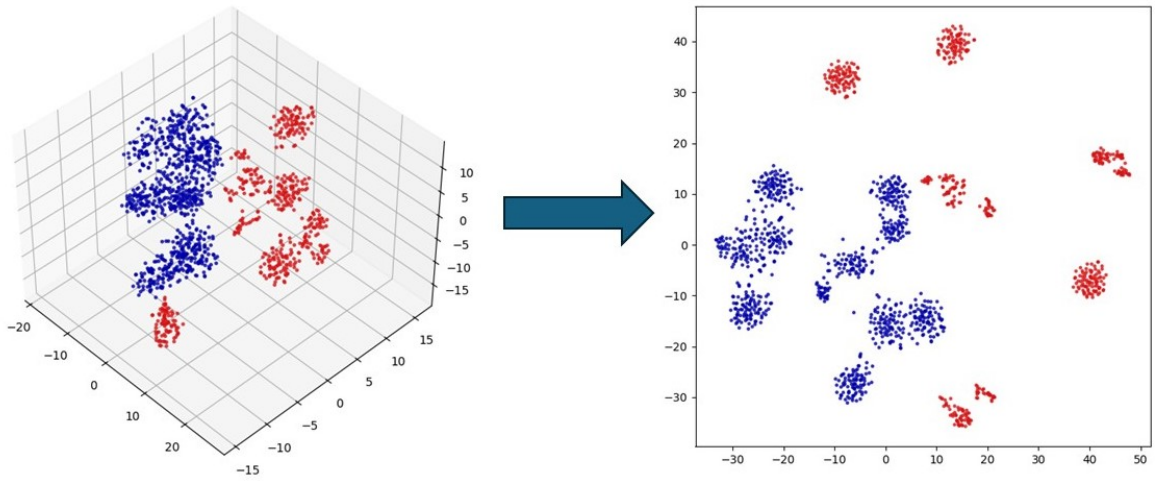


Figure 3.5: Example of using t-SNE to visualize points in a 3 dimensional space in 2 dimensions.

Let $\boldsymbol{r}_i \in \mathbb{R}^R$ denote a representation vector for a sample. t-SNE seeks a mapping

$$\boldsymbol{r}_i \mapsto \boldsymbol{z}_i \in \mathbb{R}^2 \tag{3.11}$$

such that the pairwise similarities between points in the are approximately preserved.

In the high-dimensional space, the similarity between points $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ is modeled using a conditional Gaussian distribution:

$$p_{j|i} = \frac{\exp(-\|\boldsymbol{r}_i - \boldsymbol{r}_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|\boldsymbol{r}_i - \boldsymbol{r}_k\|^2/2\sigma_i^2)}. \tag{3.12}$$

Here, σ_i is a per-sample bandwidth parameter that determines the local scale around $\boldsymbol{r}_i$. It is selected so that the perplexity of the distribution $\{p_{j|i}\}$ remains approximately constant across all i. The perplexity is a user-defined parameter that determines the effective number of nearest neighbors considered for each point. Low perplexity values focus on preserving very local structure, while higher values encourage more global organization. In practice, we set the perplexity to 30, a common choice for medium-sized datasets. In the low-dimensional space, similarities are modeled with a heavy-tailed Student-t distribution (3.13). The use of a Student-t distribution instead of a Gaussian helps to alleviate the so-called crowding problem: it allows dissimilar points to be placed far apart in the 2D map, ensuring that local clusters can spread out without overlapping excessively. Let $\boldsymbol{z}_i \in \mathbb{R}^2$ be the visualized points for t-SNE, then

$$q_{ij} = \frac{(1 + \|\boldsymbol{z}_i - \boldsymbol{z}_j\|^2)^{-1}}{\sum_{k\neq l}(1 + \|\boldsymbol{z}_k - \boldsymbol{z}_l\|^2)^{-1}}. \tag{3.13}$$

Unlike $p_{j|i}$, which is a conditional distribution, q_{ij} defines a joint probability over all pairs. The final embedding minimizes the Kullback–Leibler divergence between the two distributions [25]:

$$\mathrm{KL}(P\|Q) = \sum_i \sum_j p_{j|i} \log\left(\frac{p_{j|i}}{q_{ij}}\right), \tag{3.14}$$

so that the structure of the high-dimensional distribution P is preserved in the low-dimensional map Q. The optimization is performed over the positions of the 2D points of the t-SNE visualization, and is typically solved using gradient descent. .

In this study, we apply t-SNE to the learned representations $\boldsymbol{r}_i$ and color the resulting projections according to binarized clinical outcome (e.g., survival versus death). This allows us to visually assess whether the encoder has learned to organize the data in a clinically meaningful way.

3.4.2. Saliency Maps

Saliency maps aim to explain which parts of a model's input are most influential in shaping its internal computations. Originally introduced for image classifiers [45], the core idea is to use gradients to attribute importance to input features based on how much they affect a chosen output of the model.

In this work, we use saliency maps to gain insight into which parts of the EEG signal most influence the encoder's learned representation. An example saliency map is shown in Figure 3.6. Given an input $\boldsymbol{x}_i \in \mathbb{R}^{T \times C}$ and its corresponding representation $\boldsymbol{r}_i \in \mathbb{R}^R$, we compute the gradient of a scalar summary statistic with respect to the input using backpropagation:

$$\boldsymbol{S}_i = \left| \frac{\partial \mathcal{L}_{\text{sal}}(\boldsymbol{r}_i)}{\partial \boldsymbol{x}_i} \right|. \tag{3.15}$$

In our implementation, we explore two distinct summary statistics to quantify the influence of the input on the encoder's representation. The first, magnitude-based saliency, uses the norm of the latent representation to capture how much the input contributes to the overall activation strength. This is given by:

$$\mathcal{S}_{\text{sal}}(\boldsymbol{r}_i) = \|\boldsymbol{r}_i\| = \|f(\boldsymbol{x}_i; \Theta)\|, \tag{3.16}$$

the norm of the encoder output over all embedding dimensions. This global summary shows the influence of the signal on the overall magnitude of the latent representation for each timestep. High saliency regions may indicate time periods where the encoder detects clinically relevant signal patterns.

The second, directional saliency, projects the latent representation onto a fixed, randomly chosen direction in embedding space. This projection acts as a proxy for assessing how changes in the input influence the orientation of the encoder's output, rather than its magnitude. Formally, letting $\boldsymbol{v} \in \mathbb{R}^R$ denote a unit-norm direction vector, the summary statistic is given by:

$$\mathcal{S}_{\text{sal}}(\boldsymbol{r}_i) = \langle \boldsymbol{r}_i, \boldsymbol{v} \rangle, \tag{3.17}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. This approach reveals which parts of the input most influence the directional structure of the learned representation, offering a more geometry-aware interpretation of encoder sensitivity.

Because the saliency map is sensitive to the choice of $\boldsymbol{v}$, we compute directional saliency 50 times using randomly sampled unit vectors $\boldsymbol{v}^{(1)}, \ldots, \boldsymbol{v}^{(50)} \in \mathbb{R}^R$ with $\|\boldsymbol{v}^{(j)}\| = 1$, and take the average over all resulting saliency maps. This averaging reduces the dependence on any single direction and yields a more robust estimate of directional sensitivity.

Importantly, these saliency maps are computed with respect to the encoder alone. They do not depend on the classifier and therefore provide insight into what the encoder learns through contrastive learning alone. One can interpret the encoder as forming a summary of the EEG epoch, the saliency map then shows which parts of the input most strongly shaped certain characteristics that summary.

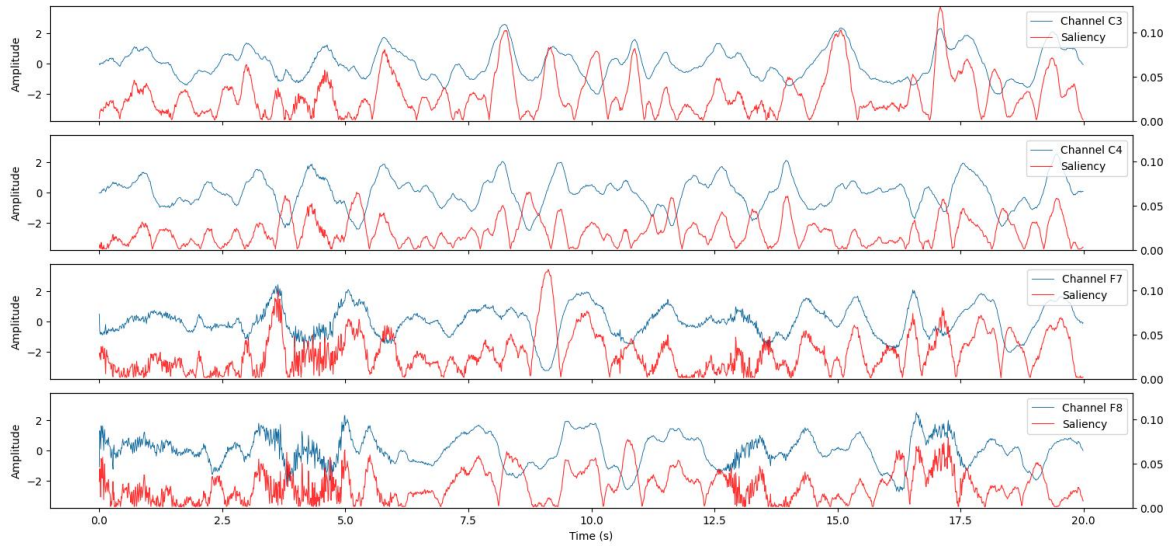


Figure 3.6: Magnitude-based saliency map for a 20-second EEG segment from a patient with PCPC score 3 after 12 months. The EEG signal is shown in blue with the scaling on the left side, the saliency map is shown in red with the scaling on the right side.

4

Classification

After training, the TS2Vec encoder is frozen and used as a feature extractor. Each EEG epoch is mapped to a fixed-length representation vector $\boldsymbol{r}_i \in \mathbb{R}^R$ by applying the encoder and max-pooling over the time axis, as described in Section 3.3. These representations serve as input features for downstream classification. Because the encoder is fixed, only the classifier is trained in the supervised stage, using labeled examples to predict long-term neurological outcome.

This section introduces the concept of classification and describes the classifier used in this thesis. In addition, Section 4.3 outlines the evaluation metrics used to assess classifier performance.

4.1. Introduction to Classification

Classification is a fundamental task in machine learning where the goal is to assign an input sample to one of several predefined categories or classes. Mathematically, an input sample is represented as an input vector $\boldsymbol{X} = (X_1, \ldots, X_p) \in \mathbb{R}^p$, where p is the number of features. For example, given a person's age, weight and height, we might classify this person as obese or non-obese. The corresponding output, or target label, is represented as $y \in \{0, 1\}$, where $y = 0$ indicates that the person is non-obese and $y = 1$ indicates that the person is obese. The predicted class label from the model is denoted $\hat{y} \in \{0, 1\}$.

Unlike contrastive learning, which is a self-supervised learning approach that learns from unlabeled data, classification is generally a supervised approach. During training, the classifier has access to labeled data, meaning that the model knows the true outcomes for each training sample resulting in the feature-label pairs $(\boldsymbol{X}^{(i)}, y^{(i)})$, where $i \in \{1, \ldots, n\}$. The aim is to learn a function $f : \mathbb{R}^p \to \{0, 1\}$ such that $\hat{y} = f(\boldsymbol{X})$ approximates the true label y as closely as possible.

In our setting, the input to the classifier is the representation vector produced by the encoder, which is denoted as $\boldsymbol{r}_i \in \mathbb{R}^R$ (that is, while $\boldsymbol{X} = \boldsymbol{r}$ in this introduction, we will use $\boldsymbol{r}$ throughout for consistency).

4.2. Our Implementation: Patient-Level Classification

We apply classification after the contrastive learning encoder has extracted structured representations from raw EEG segments. Each 20-second EEG epoch is transformed into a 320-dimensional latent vector $\boldsymbol{r}_i \in \mathbb{R}^{320}$, which serves as input to the classifier, as shown in the bottom panel of Figure 3.2. This step is essential, as raw EEG is high-dimensional, noisy, and poorly suited for direct classification. The contrastive learning stage simplifies this task by organizing the data into a feature space where semantically similar signals lie close together, enabling downstream classifiers to more easily distinguish between patients with different neurological outcomes.

Although our dataset contains over 6,000 EEG epochs, these originate from only 84 unique patients. With 5-fold cross-validation, each training set includes approximately 67 patients, substantially limiting the effective sample size for training complex models. We considered several candidate classifiers,

including logistic regression, support vector machines (SVM), neural networks, random forests, and k-nearest neighbors (k-NN). However, most of these approaches proved unsuitable in this setting. Linear models such as logistic regression performed poorly due to the nonlinear structure of the learned latent space (as evident from the t-SNE visualization in Section 5.1.1). More flexible models such as neural networks and SVMs were not pursued further because of their sensitivity to overfitting in small-sample regimes, their extensive hyperparameter tuning requirements, and relatively high computational cost.

Despite being more straightforward to tune, random forests tend to overfit when the number of feature dimensions substantially exceeds the number of independent training examples. In our case, the 320-dimensional embeddings are learned from only 64 patients per fold, making it difficult for random forests to generalize reliably in this high-dimensional, low-sample setting. Moreover, EEG is inherently low in signal-to-noise ratio, and this residual noise can persist in the learned representations, further complicating classification.

Based on these considerations, we prioritized classifiers that are non-parametric, computationally efficient, and robust under small-sample constraints. Among the remaining candidates, the k-nearest neighbors classifier consistently yielded strong validation performance with low computational cost, making it the preferred choice for this classification task.

Thus to perform classification in this latent space, we use a k-NN classifier, specifically the implementation by Scikit-Learn [34]. This method assigns a class label to a new input sample $\boldsymbol{r}_i$ by comparing it to the k most similar training samples in the feature space, its neighbors, based on a chosen distance metric. A key advantage of k-NN is that it requires no additional training, it simply stores the labeled examples and performs inference via distance comparisons. This property makes it particularly suitable for settings where the representations already encode meaningful structure, and where the dataset is limited in size.

Importantly, the k-NN classification is applied at the level of individual EEG epochs, each treated as a temporally independent sample. Each epoch corresponds to a 20-second segment of brain activity, and the classifier has no access to the temporal ordering of epochs within a patient. The final output of the model is produced by aggregating predictions across all epochs belonging to the same patient, to produce a clinically meaningful patient-level prediction. The epoch level classification and this aggregation step is explained in detail below.

Let $\mathcal{C} = \{0, 1\}$ be the set of class labels, where class 0 indicates a favorable neurological outcome and class 1 indicates a poor neurological outcome (as defined by the binarized PCPC score). For a new input epoch j of patient i, denoted as $\boldsymbol{r}_{i,j}$, we define $\mathcal{N}_k(\boldsymbol{r}_{i,j})$ as the set of its k nearest neighbors in the latent space (for a distance metric d). The predicted class is then given by:

$$\hat{y}_{i,j} = \operatorname*{argmax}_{c \in \mathcal{C}} \sum_{\boldsymbol{r}_\ell \in \mathcal{N}_k(\boldsymbol{r}_{i,j})} w_\ell \cdot 1_{[y_\ell = c]}, \tag{4.1}$$

where $1_{[y_\ell = c]}$ is the indicator function that equals 1 if y_ℓ equals class c, and 0 otherwise. The weight parameter w_ℓ is based on the distance from the input $\boldsymbol{r}_i$ to the neighbor $\boldsymbol{r}_\ell$. Usually one of the following two options is chosen:

$$w_\ell = 1 \qquad \text{(Uniform weights)} \tag{4.2}$$

$$w_\ell = \frac{1}{d(\boldsymbol{r}_{i,j}, \boldsymbol{r}_\ell)}, \qquad \text{(Distance weights)} \tag{4.3}$$

where d is a distance metric, typically Manhattan or Euclidean distance. Note that the distance metric is the same for finding the k nearest neighbors and for the weights w_ℓ.

Because our labels are binary, $y \in \{0, 1\}$, we can also compute the probability that an epoch j of patient i belongs to class 1. This gives us a probability estimate per epoch:

$$\hat{y}_{i,j}^{(\text{prob})} = \frac{1}{k} \sum_{\boldsymbol{r}_\ell \in \mathcal{N}_k(\mathbf{r}_{i,j})} w_\ell \cdot y_\ell. \tag{4.4}$$

This probability $\hat{y}_{i,j}^{(\text{prob})} \in [0,1]$ reflects how likely the model considers epoch j from patient i to belong to class 1, corresponding with poor neurological outcome.

To obtain our final predictions, we use a two-threshold strategy. First, we apply a threshold $p_1 \in [0,1]$ at the epoch level to convert the probability score $\hat{y}_{i,j}^{(\text{prob})}$ into a binary label for epoch j:

$$\hat{y}_{i,j}^{(\text{epoch})} = \begin{cases} 1, & \text{if } \hat{y}_{i,j}^{(\text{prob})} \geq p_1, \\ 0, & \text{if } \hat{y}_{i,j}^{(\text{prob})} < p_1 \end{cases} \tag{4.5}$$

Then, we aggregate the binary epoch-level predictions to obtain a patient-level score. Let m be the number of epochs for patient i. We compute the proportion of epochs classified as label 1:

$$\hat{y}_i^{(\text{mean})} = \frac{1}{m} \sum_{j=1}^{m} \hat{y}_{i,j}^{(\text{epoch})} \tag{4.6}$$

To obtain the final patient-level prediction, we apply a second threshold $p_2 \in [0,1]$:

$$\hat{y}_i^{(\text{patient})} = \begin{cases} 1, & \text{if } \hat{y}_i^{(\text{mean})} \geq p_2, \\ 0, & \text{if } \hat{y}_i^{(\text{mean})} < p_2 \end{cases} \tag{4.7}$$

This two-step thresholding process allows for independent control of sensitivity at both the epoch and patient level, which is particularly important in clinical settings where false positives must be strictly avoided. A visual summary of the aggregation strategy is provided in Figure 4.1.

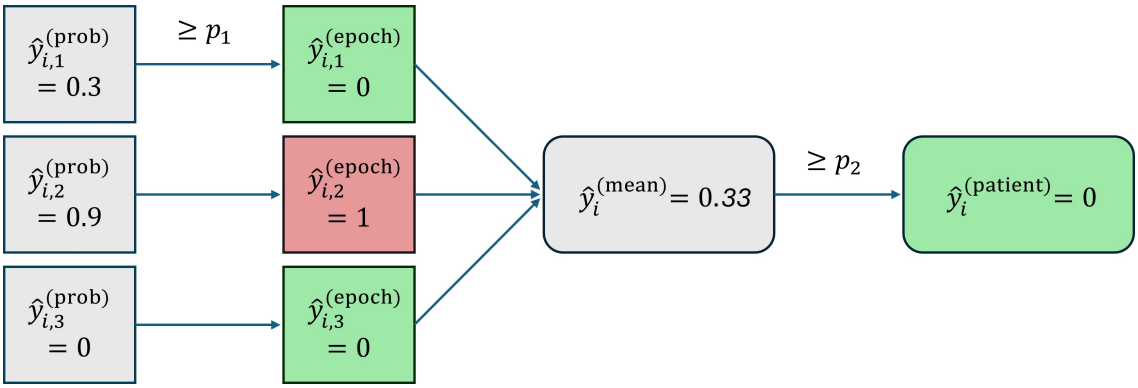


Figure 4.1: Schematic illustration of the aggregation from epoch to patient classification of patient i with 3 epochs. Here $p_1 = 0.9$ and $p_2 = 0.5$.

To determine the optimal values for the k-NN hyperparameters, such as the number of neighbors k, the choice of distance metric, and the weighting scheme, we conduct a grid search [34] using only the training data. Using training data instead of test data here is important to prevent information leakage from the test set into the model tuning process. Specifically, 5-fold cross-validation is applied on the training set. That means that the set is partitioned into five equally sized folds, and for each combination of hyperparameters, the classifier is trained on four folds and evaluated on the remaining fold. This process is repeated five times, with each fold serving once as the validation set. The hyperparameter combination that achieves the highest average AUC across these internal validation folds is selected. Table 4.1 lists the specific hyperparameter values explored during this grid search.

Once the k-NN hyperparameters have been selected, we determine the optimal classification thresholds p_1 and p_2. Using the best-performing k-NN configuration, we perform a second 5-fold cross-validation on the training data to identify the threshold values that yield the highest validation accuracy under the constraint of no false positives. To maintain a conservative classification strategy, we additionally require that $p_1 \geq 0.90$ and $p_2 \geq 0.50$. These constraints prevent the thresholds from becoming too low due to unfavorable fold splits, which could otherwise lead to false positives and

reduced precision on the test set. The final thresholds used for evaluation are computed by averaging the selected p_1 and p_2 values across the five folds.

This classifier can also be used for data with more than two label classes. In that case, the label assigned to an epoch is computed via (4.1), and the label assigned to a patient is the majority vote across epoch labels:

$$\hat{y}_i^{\text{(patient)}} = \underset{c \in \mathcal{C}}{\text{argmax}} \sum_{j=1}^{m} 1_{[\hat{y}_{i,j}=c]}. \tag{4.8}$$

This classifier can also be used for data with more than two label classes. In that case the label assigned to an epoch is the same as (4.1), but the label assigned to a patient is then the majority vote of the epoch labels of that patient, i.e.

$$\hat{y}_i^{\text{(patient)}} = \underset{c \in \mathcal{C}}{\text{argmax}} \sum_{j=1}^{m} 1_{[\hat{y}_{i,j}=c]}. \tag{4.9}$$

parameter	values
k	$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25\}$
p	$\{1, 2\}$
weights	{'uniform', 'distance'}

Table 4.1: The parameters used in the grid search for k-NN.

4.3. Evaluation Methods for Classifiers

To evaluate the performance of the classifier that is trained on the EEG epoch representations produced by the contrastive learning encoder, several well established metrics are used. These metrics quantify the model's ability to correctly classify both the positive class (class label 1, indicating a poor neurological outcome) and the negative class (class label 0, indicating a favorable neurological outcome). Since some classifiers may perform well on one metric but worse on another, it is crucial to evaluate models with multiple performance metrics [21]. This section provides an explanation of the metrics used in Section 5.

The foundation for most classification metrics is the confusion matrix, which compares the predicted labels $\hat{y}$ to the true labels y:

		Predicted class, $\hat{y}$	
		class label 0	**class label 1**
Actual class, y	**class label 0**	True Negatives (TN)	False Positives (FP)
	class label 1	False Negatives (FN)	True Positives (TP)

Table 4.2: Confusion matrix

The terms in the matrix can be explained as:

- **True Positives (TP)**: Correctly classified as positive ($y_{actual} = 1, \hat{y} = 1$)
- **False Positives (FP)**: Incorrectly classified as positive ($y_{actual} = 0, \hat{y} = 1$)
- **True Negatives (TN)**: Correctly classified as negative ($y_{actual} = 0, \hat{y} = 0$)
- **False Negatives (FN)**: Incorrectly classifier as negative ($y_{actual} = 1, \hat{y} = 0$)

Depending on the application of the classifier, different evaluation metrics become more or less important. In the healthcare domain, and especially in our study focusing on comatose patients after cardiac arrest, the consequences of misclassification are particularly severe [19]. For example, incorrectly predicting that a patient will not survive (a false positive) may lead to premature withdrawal

of life-sustaining treatment. Therefore, minimizing false positives is critical, and precision becomes a particularly important metric. As Rubinger et al. emphasize, class-specific metrics such as sensitivity, specificity and precision are essential in clinical contexts, where both false negatives and false positives carry high risk [39].

Accuracy:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$

Accuracy reflects the overall proportion of correctly classified samples, considering both surviving and non-surviving patients. This metric provides a general sense of the model performance, but it does not differentiate between the types of errors (false positives vs. false negatives).

Sensitivity (Recall):

$$\text{Sensitivity} = \frac{\text{Correctly predicted class 1}}{\text{All actual class 1}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Sensitivity measures the model's ability to correctly identify patients poor outcomes $y = 1$. A high sensitivity means that the model detects most patents who are at risk of dying.

Specificity:

$$\text{Specificity} = \frac{\text{Correctly predicted class 0}}{\text{All actual class 0}} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

Specificity measures the model's ability to correctly identify patients with a favorable outcome $y = 0$. A high specificity means that the model rarely misclassifies survivors as having poor outcomes.

Precision:

$$\text{Precision} = \frac{\text{Correctly predicted class 1}}{\text{All predicted class 1}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision reflects how reliable the model is when predicting a poor outcome $y = 1$. In our context, a high precision ensures that when the model predicts a patients death, it is likely to be correct. This is crucial medical decision making, where false alarms can result in serious consequences. Therefore, we aim for precision values close to 1, ensuring that when the model predicts death for a patient, this prediction is highly reliable.

F1 score:

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

The F1 score is the harmonic mean of precision and sensitivity. It balances the trade-off between these two metrics and is especially useful in settings with class imbalance. A high F1 score indicates that the model maintains both high precision and high sensitivity.

ROC Curve and AUC: The Receiver Operating Characteristic (ROC) curve plots the true positive rate (sensitivity) against the false positive rate, $\frac{\text{FP}}{\text{FP}+\text{TN}}$, across a range of classification thresholds. The Area Under the Curve (AUC) provides a scalar measure of overall performance, where a perfect classifier scores 1.0, and a random guessing classifier, e.g. flipping a coin for either class 0 or 1, scores 0.5. A model with a higher AUC is generally considered better at distinguishing between the two classes. Figure 4.2 illustrates this. For our (patient-level) classifier, the ROC curves are generated by varying the threshold, $p_{threshold}$.

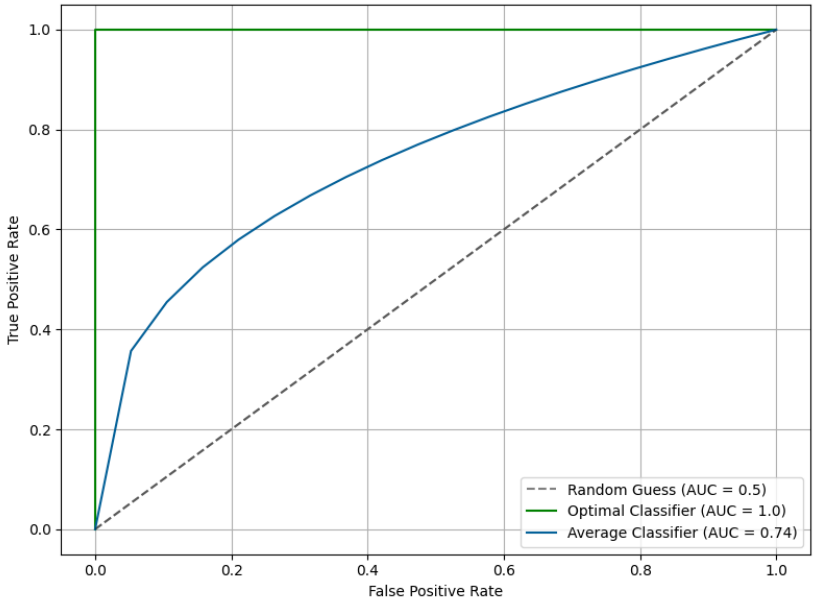


Figure 4.2: Plot with an example ROC curve (blue), the optimal ROC curve (green) and the random guess line (gray). The AUC is shown in the bottom right.

In Section 5, we report all these metrics for the classifier predictions.

4.4. Prior Methods for EEG Classification

We compare our contrastive-learning pipeline with existing research on pediatric EEG prognostication after cardiac arrest and with high-performing[1] contrastive learning for multivariate time series methods for which reproducible code is publicly available.

Specifically, we included: the only published pipeline that targets pediatric post-cardiac-arrest EEG directly (Hunfeld et al.), three generic contrastive learners for multivariate time series with open code bases (TS-TCC, CA-TCC and COMET), and two earlier clinical baselines that rely on expert-engineered features (Fung et al. and Lee et al.)

None of the contrastive frameworks returns patient-level labels out of the box. For a fair head-to-head comparison we therefore added a simple majority-vote aggregation over the 20-s epochs belonging to the same child. This step was applied identically to TS-TCC, CA-TCC and COMET, it is explicitly mentioned below where relevant. This modification may have improved their performance, particularly in clinical settings where patient-level predictions are more meaningful than per-epoch classifications, however we will not take this into account. Apart from this alteration, and of course training the models on our data, no changes were made to the models or their (hyper)parameters.

We do not include direct comparisons with the models of Fung et al. or Lee et al., as their implementations are not publicly available. Furthermore, both studies report lower performance on their own datasets, despite being similar in size and patient population, than the model presented by Hunfeld et al., whose approach we do compare to. Fung et al. reports results on a cohort of 89 pediatric patients, while Lee et al. use 69 patients. Our dataset contains 84 patients, making these benchmarks comparable in scale. We include these studies in the discussion as they are, to our knowledge and together with Hunfeld et al., the only other works that develop models for neurological outcome prediction from early EEG after pediatric cardiac arrest.

Fung et al. In a prospective cohort of 89 children whose EEG was started a median 6.9 h after return of circulation, Fung et al. [14] scored three visual descriptors—background category, stage-2

[1] "High-performing" refers to methods that rank at, or near, the top on at least three public time-series leaderboards, according to their original authors.

sleep transients and reactivity/variability—using the ACNS terminology. A logistic-regression model that combined those three variables achieved an AUC of 0.84 for 12-month mortality and a specificity of 0.97. Although the method is highly specific, it depends on labor-intensive expert annotation and produces only epoch-level predictions.

Lee et al. Lee et al. [27] analysed 69 of 87 consecutively resuscitated children who had both early (0–17 h) and late ($\geq$ 18 h) recordings. From 5-min artifact-free epochs they extracted eight qEEG features plus age and trained a random-forest classifier. For the late-EEG window, which is most comparable to our 24-h protocol, the model reached an accuracy of 0.70 and an AUC of 0.74, with a sensitivity of 0.70 and a specificity of 0.62 . Patients who died from non-neurologic causes were excluded, and no patient-level aggregation was implemented.

Hunfeld et al. Using the same raw dataset as the present study (84 usable recordings at 24 h), Hunfeld et al. [19] computed 27 qEEG features from a 30-min segment and trained a random forest. The model obtained an AUC of 0.90 and an accuracy of 0.77 at the patient level, with perfect specificity and precision (1.00) for death prediction. Continuity and amplitude were the dominant features in this research. These results underscore the prognostic utility of engineered qEEG features and provide a strong baseline for comparison with our contrastive learning approach, which operates directly on raw EEG.

TS-TCC TS-TCC [13] is a supervised contrastive learning framework for time series classification. The method introduces a dual-view training setup in which each input sequence is augmented twice to create positive sample pairs, and the model learns to minimize the distance between representations of the same input while maximizing distance from other inputs. Unlike our approach, TS-TCC is trained using labeled data, combining contrastive representation learning with supervised classification loss. After pretraining, a classifier head is fine-tuned on the labeled training set. We implemented TS-TCC and its variant CA-TCC (which introduces class-aware sampling of negative sample pairs) on our own dataset, and extended both methods with a patient-level aggregation step. This was necessary to produce patient-level predictions, which were not part of the original pipeline. On our data, TS-TCC achieved a mean accuracy of 0.788, AUC of 0.854, and precision of 0.917. CA-TCC showed slightly lower performance, with an accuracy of 0.775 and AUC of 0.847.

COMET COMET [49] is a hierarchical, self-supervised contrastive learning approach developed specifically for clinical time series. The model learns patient-level representations by aggregating features from multiple time windows belonging to the same individual. During training, it uses a contrastive loss that pulls together augmented segments from the same patient and pushes apart those from different patients. This architecture integrates patient-level information directly into the learning objective, rather than applying aggregation as a post hoc evaluation step. We trained COMET on our dataset and adapted the output layer to enable binary classification of patient outcome like with TS-TCC and CA-TCC. On our data, COMET achieved an accuracy of 0.798, AUC of 0.836, and F1 score of 0.821. These results provide a strong benchmark for contrastive learning methods that incorporate structural information about the patient during training.

5

Results

In this section, we evaluate the performance of the full classification pipeline. We begin with a qualitative assessment of the encoder representations using t-SNE and saliency analysis as explained in Section 3.4. This helps us understand whether the model has learned clinically meaningful structures from the EEG data. We then report quantitative results for patient-level classification performance using the evaluation metrics introduced in Section 4.3.

The model configuration used in all experiments consists of an encoder trained on the training set (see Section 3.3), combined with a k-Nearest Neighbors classifier, also trained using the training set (see Section 4.2), where the hyperparameters of the k-NN and the thresholds p_1 and p_2 are chosen using internal 5-fold cross-validations on the training data. The classification results are given on the test data, as the mean and standard deviation over the folds.

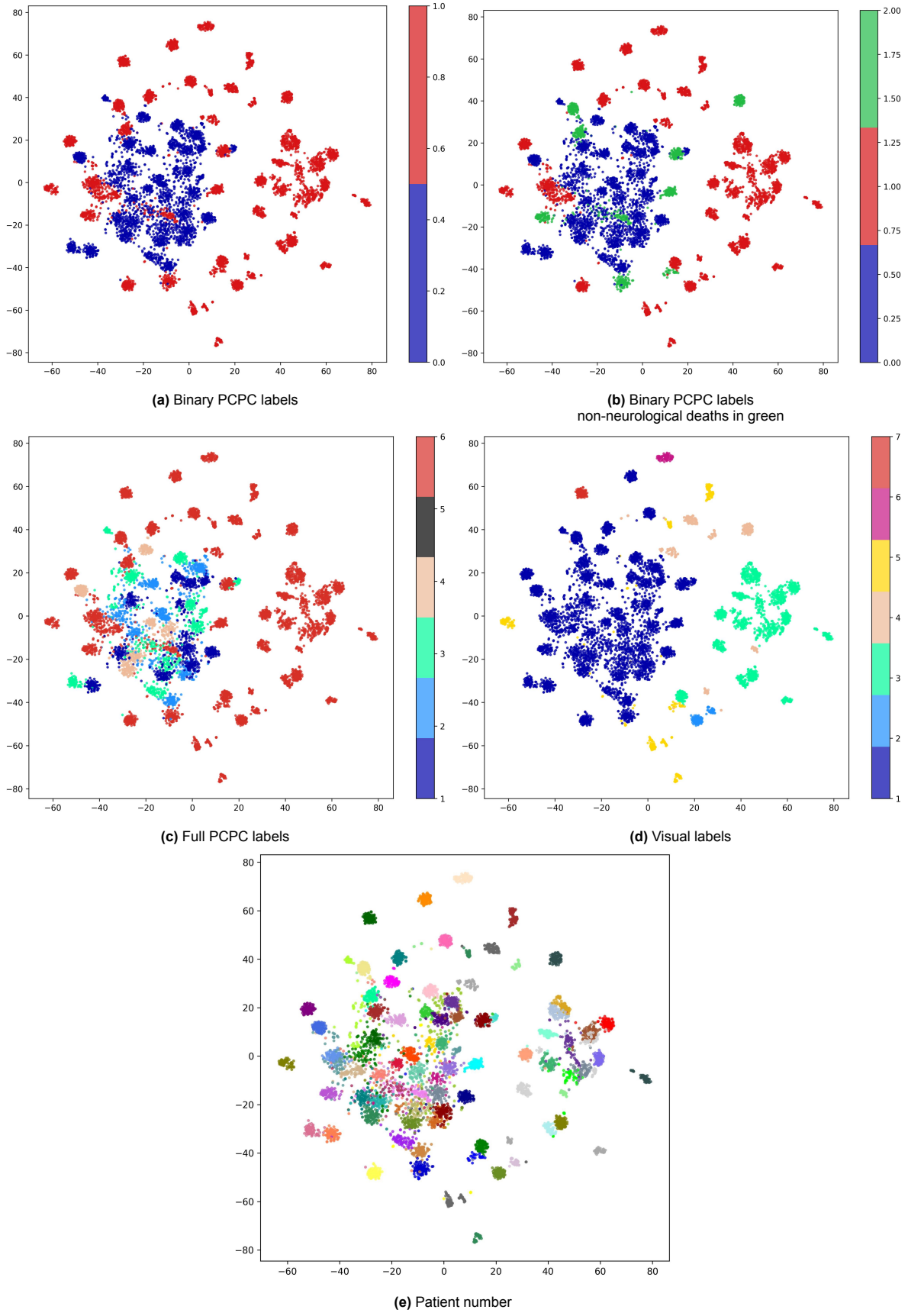
5.1. Encoder Representations

To better understand what information is captured by the encoder, we qualitatively analyze its learned representations. We focus on two complementary techniques: t-SNE for visualizing global structure in the latent space, and saliency maps for inspecting which parts of the input most influence the encoder's output.

5.1.1. t-SNE Analysis

To qualitatively assess the structure of the learned representations, we apply t-SNE to the 320-dimensional output vectors produced by the encoder as explained in Section 3.4.

Figure 5.1 shows the resulting 2D embeddings, with different label overlays, for one of the five folds. Note that these labels were added afterwards for us to interpret the results from the t-SNE, the encoder does not have access to the labels and is purely trained using the contrastive learning objective. Furthermore, while the encoder is not trained using the test data, in the t-SNE visualization we use the encoder to visualize all 6669 epoch (so both the train and the test data). The t-SNE plots of the remaining four folds are shown in Appendix B. These plots reveal that the representation spaces are qualitatively similar across folds, suggesting that the encoder generalizes consistently across different training splits.



(a) Binary PCPC labels

(b) Binary PCPC labels
non-neurological deaths in green

(c) Full PCPC labels

(d) Visual labels

(e) Patient number

Figure 5.1: Two-dimensional t-SNE embeddings of encoder representations. Each point corresponds to a single EEG epoch and is colored by the associated labels for that patient. The PCPC labels are the PCPC score after 12 months.

Figure 5.1e shows that EEG epochs from the same patient tend to cluster together in the latent space. This suggests that the encoder has captured some patient-level structure in the data, assigning similar representations to temporally different recordings from the same individual. This is consistent with expectations, as comatose patients often show relatively stable EEG patterns over time.

In Figure 5.1a, we observe a qualitative trend in which epochs from patients with favorable neurological outcomes, $y = 0$ (PCPC 1–4), appear more frequently in a localized region of the embedding. However, there is substantial overlap between outcome groups, particularly among the unfavorable outcomes, $y = 1$ (PCPC 6), which are broadly dispersed. This may indicate that the encoder captures some structure correlated with outcome, although not in a linearly separable way.

Figure 5.1b highlights patients who died from non-neurological causes (shown in green). These points are scattered throughout the embedding space, and overlap notably with both outcome classes. Interestingly, they appear to cluster more closely with patients who survived, rather than with those who died from neurological injury. One possible explanation is that these patients exhibited EEG patterns that resembled those of neurologically intact individuals, despite ultimately dying from other causes. This interpretation is supported by the visual EEG labels: with one exception, all of these patients received a visual label of 1, corresponding to a continuous background with normal amplitudes. This suggests that the encoder captures some of the signal structures that align with expert assessments of preserved cerebral function, independent of survival outcome.

In Figure 5.1c, the full PCPC labels provide a more granular view (see Table 2.1). Again, while patients with scores 1–4 (survivors) are somewhat localized, patients with score 6 (non-survivors) appear scattered across the embedding. No clear substructure among PCPC 1–4 is observed, suggesting that the encoder does not sharply differentiate between the EEG epochs of patients with different levels of impairment within the survivor group. This may reflect the difficulty of capturing subtle cognitive and functional differences from early EEG alone. Notably, even expert visual assessment cannot always reliably distinguish between these outcome categories, underscoring the inherent difficulty of prognosticating fine-grained neurological outcomes from early EEG.

Finally, Figure 5.1d visualizes the t-SNE projection using expert-labeled EEG background patterns (see Table 2.2). Here, although not perfect, distinct clusters are visible, especially for labels 1 and 3 (continuous normal activity and electrocerebral silence). This suggests that the encoder has learned to separate major EEG patterns, supporting the hypothesis that the encoder representations are sensitive to clinically meaningful electrophysiological structure, even if outcome separation remains partial.

Taken together, these plots provide partial support for the hypothesis that the encoder captures clinically meaningful variation. EEG epochs from the same patient cluster closely, indicating that patient identity is well preserved in the latent space. Some visually assessed EEG background types (i.e. label 1 and 3) also show clustering, particularly in classes with a larger number of patients. However, for less frequent EEG types, no clear separation can be observed, either due to insufficient data or due to the encoder's inability do differentiate between the different EEG background patterns. The overlap between outcome groups and the absence of strong cluster separation indicate that neurological outcome cannot be inferred directly from the individual embeddings. This aligns with the clinical complexity of post-arrest EEG and underscores the need for downstream classification to aggregate evidence across time and context.

5.1.2. Saliency Analysis

While t-SNE provides a global view of the encoder's latent space, with saliency maps we want to inspect how specific input regions affect the learned representations on an individual-epoch level. In this analysis, we visualized saliency maps for the two types of summary statistics discussed: magnitude-based saliency, which measures sensitivity with respect to the norm of the latent vector, and directional saliency, which probes the input's influence on the orientation of the representation in embedding space.

Figure 5.2 shows an example of a magnitude-based saliency map for one EEG epoch. We observe that regions of large amplitude (both positive and negative) tend to yield stronger saliency, which is expected given the nature of the norm-based summary statistic. This is confirmed quantitatively: the

correlation between the saliency signal and the absolute value of the EEG signal is high with a mean correlation of 0.86 and a standard deviation of 0.05 across the four different EEG channels (the correlations are $[0.79, 0.92, 0.86, 0.86]$ for channels C3, C4, F7, F8 respectively). This strong correlation is consistent with the definition of the norm-based summary statistic, which emphasized the magnitude of the representation vector. However, it also limits the interpretability of this type of saliency map, as it mostly reflects signal energy rather than more nuanced or abstract features.

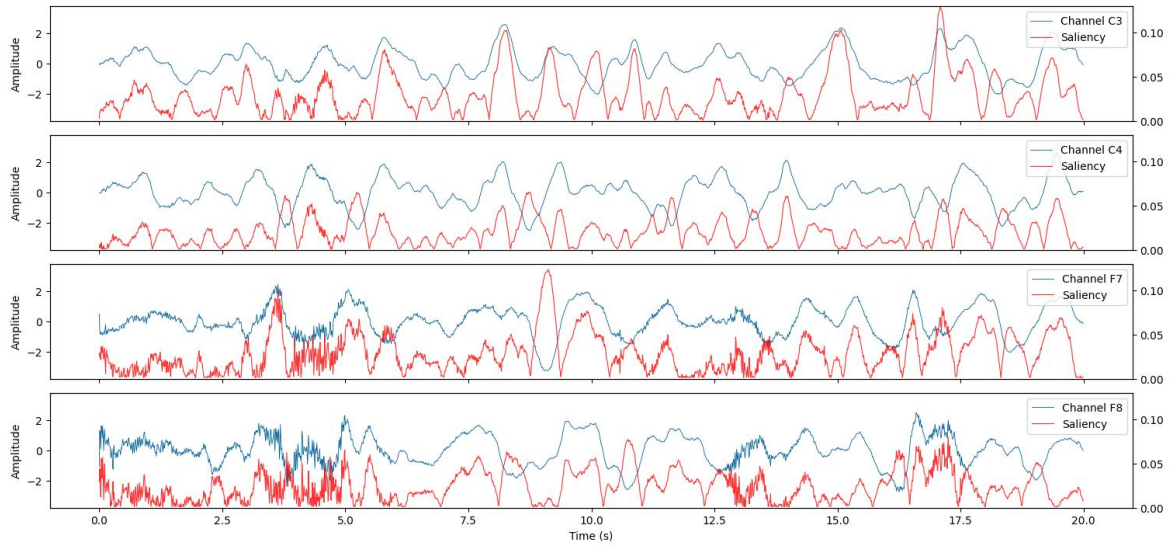


Figure 5.2: Magnitude-based saliency map for a 20-second EEG segment from a patient with PCPC score 3 after 12 months. The EEG signal is shown in blue with the scaling on the left side, the saliency map is shown in red with the scaling on the right side.

Directional saliency maps are shown in Figure 5.3, they highlight which parts of the input most influence the direction of the encoder output in latent space, rather than its magnitude. Visually, the saliency patterns are less intuitive than those seen in magnitude-based maps, and the saliency values vary substantially across channels. Especially channel C3 and C4 seem less intuitive, while channel F7 and F8 seem to follow the EEG signal more closely. Unlike magnitude saliency, the correlation between directional saliency and the EEG signal is moderate with a mean correlation of -0.06 and a standard deviation of 0.43 over the four channels (the correlations are $[0.16, -0.53, 0.54, -0.40]$ for channels C3, C4, F7, F8 respectively). This indicates that these maps capture different and less easily interpretable aspects of the input. Nonetheless, directional saliency offers a complementary perspective on encoder behavior, potentially revealing more abstract features not directly ties to signal amplitude.

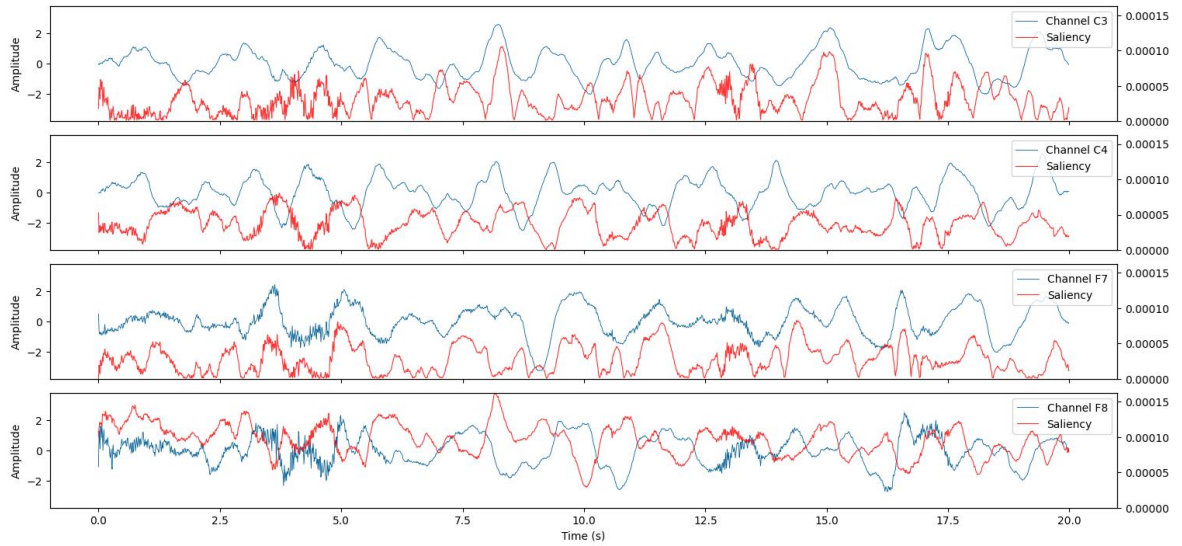


Figure 5.3: Directional-based saliency map for a 20-second EEG segment from a patient with PCPC score 3 after 12 months. The EEG signal is shown in blue with the scaling on the left side, the saliency map is shown in red with the scaling on the right side.

In summary, saliency maps suggest that the encoder is sensitive to temporal patterns in the EEG. However, their interpretation remains highly dependent on the choice of summary statistic and is not directly linked to patient outcome. Accordingly, we include them as an exploratory tool rather than as a basis for conclusions. Additional examples are provided in Appendix C.

5.2. Classifier Results

The final classification model predicts long-term neurological outcome at the patient level based on EEG representations produced by the TS2Vec encoder. The evaluation follows the procedure outlined in Section 4.3, and both the mean and the standard deviation is reported over the five cross-validation folds.

The full confusion matrix is shown in Figure 5.4, here the numbers of the folds are summed to give a view of the performance on the full dataset. Across the five folds, the classifier achieved a total 37 true positives, 29 true negatives, 0 false positives, and 18 false negatives. The absence of false positives means that no patients who survived were incorrectly classified as deceased.

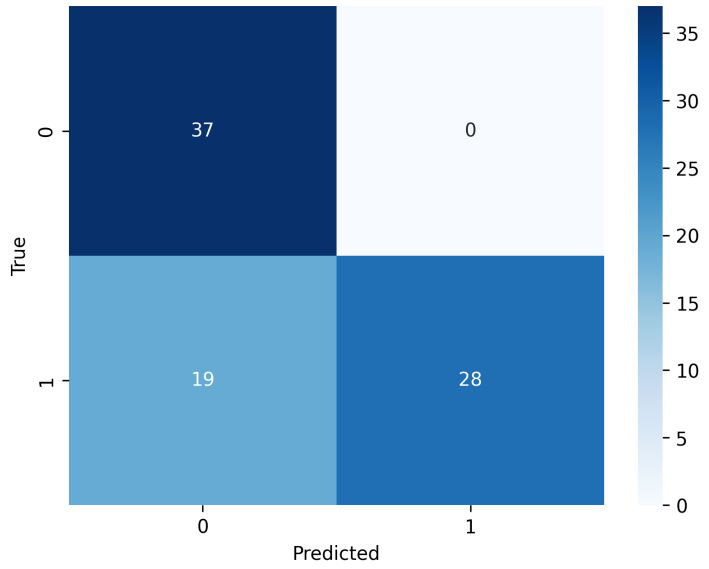


Figure 5.4: Confusion matrix of the classifier with the binary PCPC labels at 12 months after cardiac arrest, aggregated over five cross-validation folds.

Table 5.1 summarizes the performance across key classification metrics. The classifier achieves a mean accuracy of 0.774, with perfect specificity and precision, both of 1.000. This indicates that the model never predicts a poor neurological outcome for a patient who in fact had a favorable outcome. Clinically, this is crucial as such false positives may lead to premature withdrawal of life-sustaining treatment. The perfect precision ensures that every patient predicted to have a poor outcome indeed had one, while the perfect specificity confirms that all patients who survived were correctly identified. These properties make the classifier especially well-suited for high-stakes clinical scenarios where overestimating risk carries serious consequences.

Sensitivity is lower at 0.581, indicating that the model fails to detect a substantial amount of patients who died. This reflects a conservative classification strategy that prioritizes the elimination of false positives, even at the cost of missing some true positives. Notably, this overall sensitivity masks a difference between patients who died from neurological causes (38) versus non-neurological causes (9). Among the 38 patients who died due to neurological injury, the model correctly predicts an unfavorable outcome in 26 cases, yielding a sensitivity of 0.684. By contrast, only 2 out of 9 patients who died from non-neurological causes are correctly classified, resulting in a sensitivity of 0.222 for this subgroup. The resulting F1 score for the total cohort was 0.733, and the AUC is 0.861, indicating good overall discriminative ability despite the cautious decision boundary.

Metric	score
Accuracy	0.774 ± 0.045
AUC	0.861 ± 0.092
Sensitivity (Recall)	0.581 ± 0.070
Specificity	1.000 ± 0.000
Precision	1.000 ± 0.000
F1 score	0.733 ± 0.057

Table 5.1: Classification results on the test set. Each metric reports the mean score and standard deviation across five cross-validation folds.

The ROC curve is shown in Figure 5.5, with the solid blue line representing the mean ROC across folds and light blue lines representing the individual folds. The curve lies well above the diagonal line corresponding to random guessing, with a high initial value indicating that many true positives are

detected even at very low false positive rates. This is consistent with the model's perfect specificity and precision. The ROC curve was generated by varying the threshold p_2.

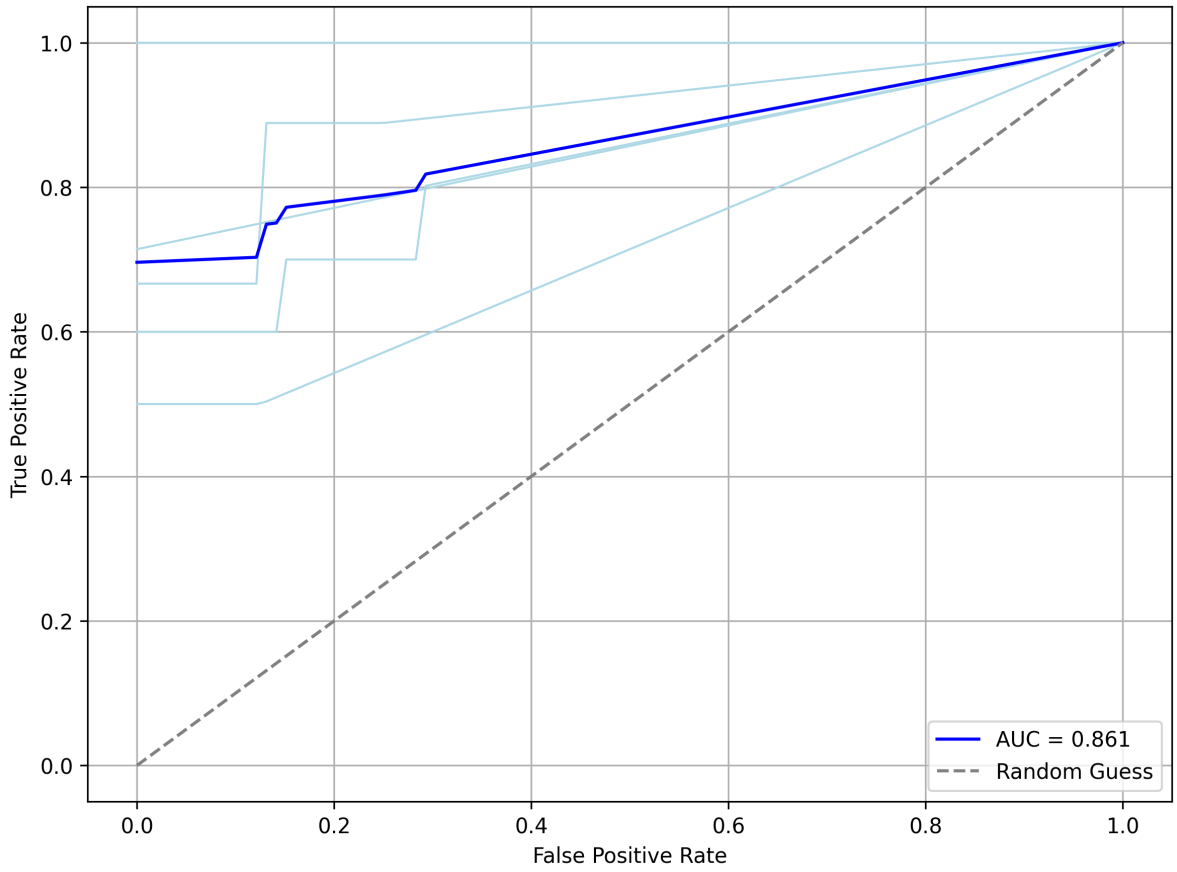


Figure 5.5: The ROC curve of the classifier. The blue line indicates the mean curve over the 5 folds, the light blue lines are the 5 folds. The varying variable is p_2.

These quantitative results are consistent with the earlier t-SNE visualizations in Section 5.1.1, where only partial separation between both outcome groups was observed in the latent space. The classifier's strong performance, particularly in specificity and precision, demonstrates effective separation between certain outcome groups in the latent space. These findings are explored further in the conclusion and discussion.

5.2.1. Results with Full PCPC Scale

When the outcome task is expanded from a binary label to the full six-level PCPC scale the classifier's performance deteriorates substantially (see Figure 5.6). The classifier still correctly identifies 32 out of 47 patients with PCPC score 6 (deceased), which is three more than in the binary setting. This improvement is likely due to the fact that the epochs labeled as surviving are now distributed across four outcome categories, thus losing the majority-vote advantage they previously held in binary classification. In contrast, only 6 of the 37 surviving patients are assigned their exact PCPC sore, an accuracy of 0.162. Five patients with PCPC score 1 and one patient with PCPC score 4. No patients are ever classified correctly as PCPC 2 or 3.

These misclassifications can be attributed to both data imbalance and the clinical complexity of the PCPC scale. More than half of the patients in the dataset have a PCPC score of 6, while some of the the classes contain fewer than ten samples. As a result, the k-Nearest Neighbors classifier is biased towards the dominant class. In addition, the electrophysiological differences between PCPC scores 1 to 4 are likely to be much more subtle than the contrast between survival and death, making them harder to distinguish based on early EEG. Together, these factors explain the decline in performance

for the multiclass setting. Although a few additional mortality cases are correctly identified, this comes at the cost of a substantial drop in classification accuracy among survivors.

One interesting take on these results is interpreting the predictions from the patient's perspective: given a predicted PCPC score $\hat{y}$, what is the likelihood that the actual outcome is no worse than predicted, i.e., $y_{\text{true}} \leq \hat{y}$? This corresponds to treating the model's output as an upper bound on outcome severity. The overall accuracy under this criterion is 0.738, indicating that the model's predictions are reasonably consistent with a conservative estimation strategy, although there are still many cases where the predicted outcome differs notably from the true score.

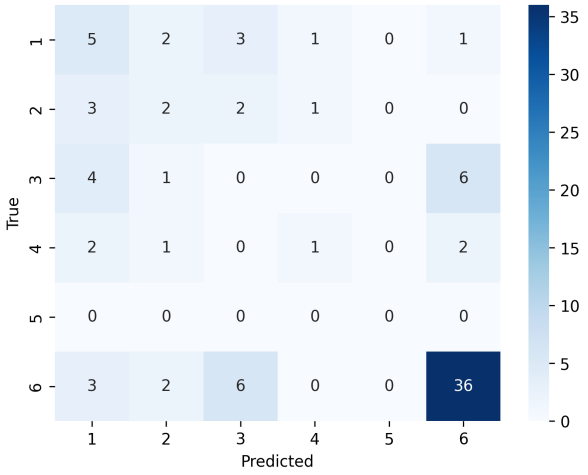


Figure 5.6: Confusion matrix of the classifier with the full PCPC labels at 12 months after cardiac arrest, aggregated over five cross-validation folds.

5.2.2. Results with Visual Labels

To verify whether the learned representations encode classical EEG background patterns, we repeated the classifier evaluation using the visual labels that clinical neurophysiologists assigned at 24 hours after return of circulation (see Section 2.4.2). Because there labels provide a description of the cerebral activity, good performance here would indicate that the encoder captures neuro-physiological structure rather than merely outcome-related artifacts.

Binary Setting: Normal Continuous Background Versus All Other Patterns We first collapsed the seven original classes into a binary task that singles out label 1 (continuous background with normal amplitudes, $\geq 20\mu V$) from the remaining patterns. For this task of predicting binary visual background patterns, we remove the constraints on the thresholds p_1 and p_2. This task differs from clinical outcome prediction in that it does not carry the same ethical or treatment implications. Thresholds p_1 and p_2 are therefore purely selected based on accuracy, rather than being constrained to enforce conservative predictions.

The resulting confusion matrix is shown in Figure 5.7. The corresponding performance metrics are given in Table 5.2, here we see a high performance across all different measures. These numbers imply that the encoder-classifier pipeline recognizes a physiologically intact, continuous background with near certainty. This finding mirrors the label-outcome distribution in our dataset where all patients who survived to 12 months were assigned visual label 1 (see Figure 2.8). Accordingly, the binary task is largely equivalent to detecting a continuous, normal-amplitude background at 24 hours, which is an EEG pattern that is generally interpreted as favorable, rather than making a direct inference about long-term neurologic status. The ROC curve is given in Figure 5.8, which again shows very good performance, indicating that the classifier consistently distinguishes between normal and abnormal background patterns across folds. Note that two folds have a perfect ROC (a true positive rate of 1.0 everywhere), which is why only four light blue lines are visible in the plot.

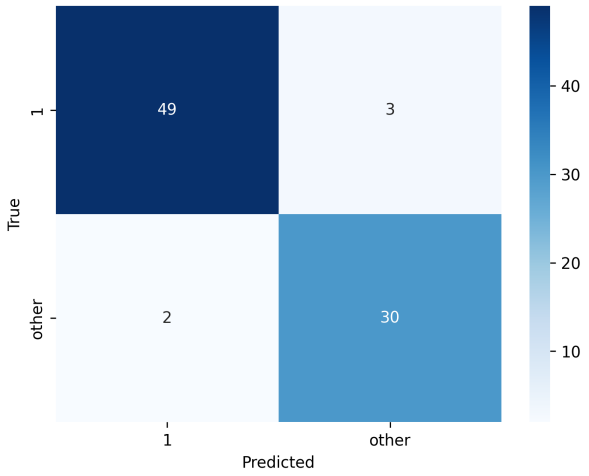


Figure 5.7: Confusion matrix of the classifier with binary visual labels, aggregated over five cross-validation folds.

Metric	score
Accuracy	0.941 ± 0.053
AUC	0.962 ± 0.043
Sensitivity (Recall)	0.951 ± 0.061
Specificity	0.944 ± 0.074
Precision	0.921 ± 0.102
F1 score	0.932 ± 0.064

Table 5.2: Classification results on the test set for the binary visual labels. Each metric reports the mean score and standard deviation across five cross-validation folds.

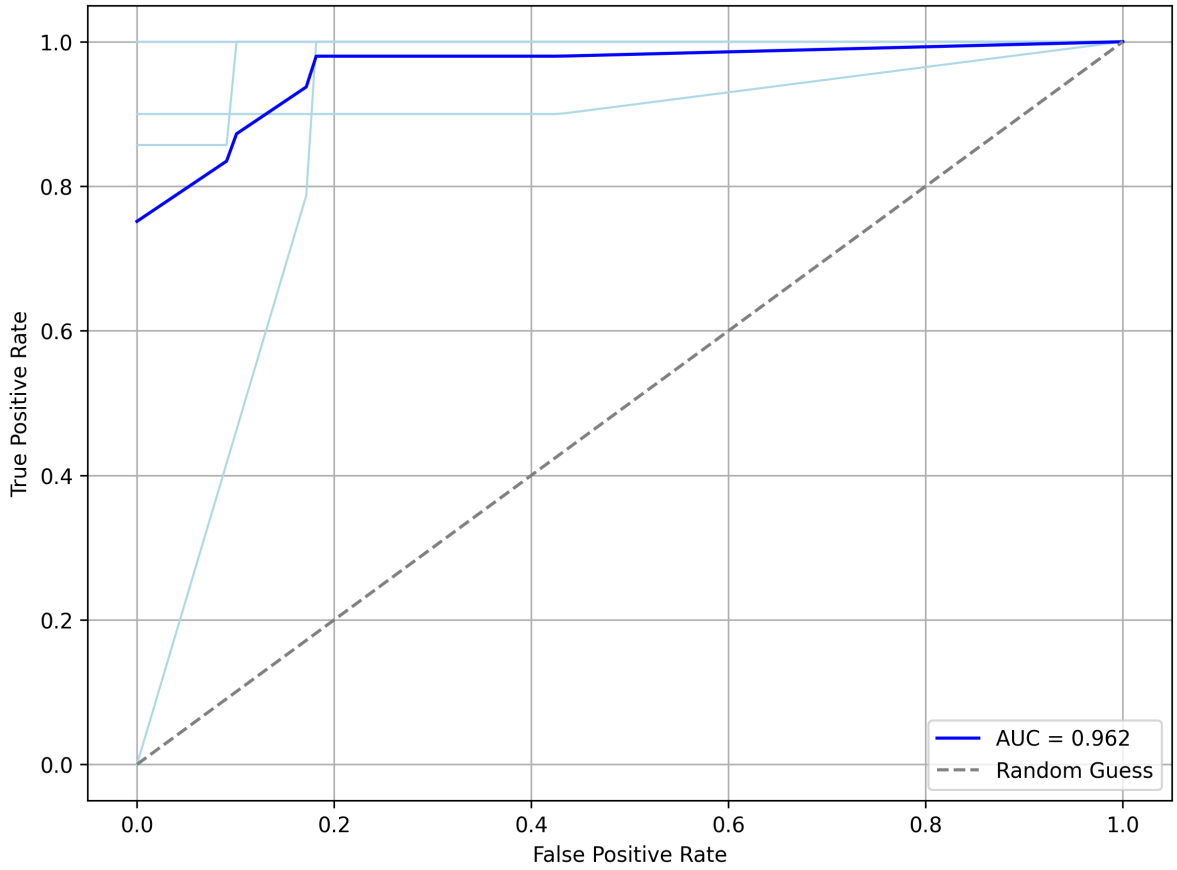


Figure 5.8: The ROC curve of the classifier on binary visual labels. The blue line indicates the mean curve over the 5 folds, the light blue lines are the 5 folds. The varying variable is p_2.

Multiclass Setting: Full Spectrum of Background Patterns The classification results on the full 7 labels (see Figure 5.9) paints a more nuanced picture. We see that labels 1 and 3 achieve very good performance with 51 out of 53 patients with label 1 (continuous background pattern with normal amplitudes, $\geq 20\mu V$) being correctly classified and 17 out of 18 patients with label 3 (electrocerebral silence) being correctly classified. The remaining suppressed and periodic classes often overlap. Labels 2 and 4-6 are frequently mistaken for label 1, while labels 6 and 7 are confused with each other. The macro-averaged accuracy (averaging the accuracy of each class, giving equal weight to each class regardless of the number of instances in that class) therefore drops to about 0.71.

There are two factors that likely contribute to these findings: the class imbalance and the ambiguous morphology of EEG signals. More than half of all labeled epochs belong to label 1, whereas some patterns appear in fewer than ten patients. With such skewed data, the k-NN decision boundary is biased towards the dominant classes, 1 and 3. Secondly, because of the ambiguous morphology of EEG signals, the distinction between burst-suppression variants or low-amplitude suppression can be subtle and subjective, even for human experts. Limited inter-rater reliability therefore sets an upper bound on achievable performance.

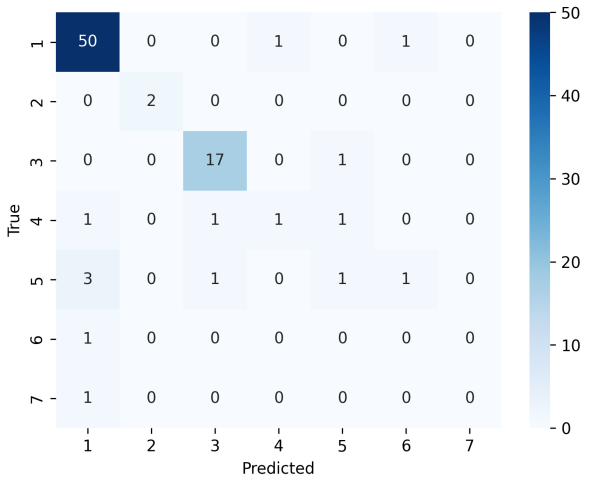


Figure 5.9: Confusion matrix of the classifier with full visual labels, aggregated over five cross-validation folds.

Our results indicate that the encoder arranges its latent space so that the two extreme background patterns (normal continuous background pattern and electrocerebral silence) are clearly separated, even though the model was never shown any EEG labels during training. The clear correspondence between visual label 1 and favorable outcome also explains why the binary outcome classifier in Section 5.2 attains perfect precision: once the pipeline detects a normal continuous background, it rarely produces an unfavorable prognosis. Conversely, the difficulty in disentangling the rarer abnormal patterns highlights a data need. Additional examples may be required before the method can support fine-grained EEG interpretation.

In summary, the visual-label experiment supports the validity of the learned representations. They encode clinically meaningful information about background continuity and amplitude that aligns with expert assessment of background EEG patterns.

5.2.3. Comparison to Prior Work

To place our results in a broader context, we benchmark our pipeline against a selection of strong publicly available time-series classifiers on the identical 84 patient cohort (see Section 4.4). The set spans three different methodologies:

- a qEEG, feature-engineering pipeline devised specifically for pediatric EEG (Hunfeld et al [19]),
- a hierarchical contrastive framework for general medical time series that outputs epoch-level representations (COMET [49]), and
- two generic contrastive learners for general multivariate time series (TS-TCC and its class-aware variant CA-TCC [13]).

Because COMET, TS-TCC and CA-TCC return one prediction per epoch, we appended a majority-vote aggregation so that all methods produce patient-level labels before comparison.

Metric	Our model	Hunfeld et al.	COMET	TS-TCC	CA-TCC
Accuracy	0.77	0.77	**0.80**	0.79	0.78
AUC	0.86	**0.90**	0.84	0.85	0.85
Sensitivity (Recall)	0.58	0.60	**0.83**	0.65	0.63
Specificity	**1.00**	**1.00**	0.76	0.94	0.94
Precision	**1.00**	**1.00**	0.81	0.92	0.92
F1 score	0.73	0.75	**0.82**	0.75	0.74

Table 5.3: Patient-level performance of competing methods (mean over five cross-validation folds).

Table 5.3 highlights two contrasting performance profiles. Our unsupervised model and the engineered-feature (qEEG) approach of Hunfeld et al. both operate with perfect specificity and precision of 1.00. In other words, neither model ever misclassifies a child with favorable long-term outcome as unfavorable in this cohort. This conservative behavior is attractive in a prognostic setting here an unjustifiably pessimistic prediction could contribute to premature treatment withdrawal.

At the opposite end of the spectrum, COMET prioritizes sensitivity more. It identifies the largest proportion of poor outcome (sensitivity 0.83) and therefore yields the highest overall accuracy of 0.80, albeit at the cost of a moderate false-positive rate (specificity 0.76). The generic contrastive learners TS-TCC and CA-TCC occupy an intermediate position, balancing sensitivity and specificity between these two extremes. Consequently, the preferred model depends on clinical priorities, whether the aim is to minimize false alarms or to maximize early detection of unfavorable prognosis.

6

Conclusion

This study shows that a self-supervised TS2Vec encoder, trained on 6,669 unlabeled EEG epochs from 84 comatose children, can serve as the foundation for a clinically conservative outcome-prediction pipeline. Combined with a k-nearest neighbor classifier, the representations yield an accuracy of 0.774 ± 0.045 and an AUC of 0.861 ± 0.092 at the patient level while maintaining perfect specificity and precision. In other words, the model never predicts death for a child who ultimately survived. This property is critical in a pediatric intensive-care setting where an unjustifiably pessimistic forecast could influence withdrawal of life-sustaining therapy. While this binary setting provides a strong foundation, extending the task to the full six-level PCPC scale resulted in a marked drop in per-class accuracy, particularly among survivors. Nevertheless, the binary performance demonstrates that the learned representations support reliable patient-level outcome prediction.

The latent space separates the two extreme EEG background patterns, continuous normal activity and no cerebral activity, without having seen any explicit EEG labels during training. Strong performance on the binary visual-label task, together with t-SNE visualizations that cluster epochs from the same patient and distinguish the principal background classes, suggests that the encoder has indeed captured neurophysiological structure rather than overfitting to outcome-related artifacts. These findings confirm that contrastive learning alone can recover physiologically meaningful structure from raw EEG.

Compared with established baselines, our pipeline matches the perfect specificity of the engineered qEEG model by Hunfeld et al. while dispensing with manual feature design, and achieves the second best AUC. however, sensitivity remains moderate at 0.581 ± 0.070, particularly for patients who died from non-neurological causes.

In summary, contrastive learning on raw pediatric EEG can produce specific, clinically meaningful predictions of long-term neurological outcome. The approach offers an alternative to feature-engineered pipelines and merits further investigation as an adjunct tool in neuro-prognostication after cardiac arrest.

7

Discussion

This study demonstrates that a self-supervised encoder, trained on raw EEG without outcome supervision, can support a conservative classification pipeline for prognostication after pediatric cardiac arrest. While the results are promising, a number of methodological, clinical, and interpretability-related considerations must be addressed to contextualize the findings and guide future work.

The encoder appears to capture physiologically meaningful structure. Visualizations using t-SNE show that the latent space clusters EEG epochs by patient identity and separates extreme background patterns, such as continuous activity and suppression. This aligns with clinical expectations and suggests that the encoder organizes the data in a way that reflects underlying brain function. However, the representations were not separable with respect to long-term neurological outcome. Patients with PCPC scores of 1–4 overlapped considerably in latent space, possibly due to small class sizes but also likely because EEG differences between mild and moderate impairment are subtle or absent in the first 24 hours. These findings reinforce the need for a downstream classifier to map these representations to clinical labels.

The classification component achieved perfect specificity and precision, indicating that the model never predicted a poor outcome for a patient who ultimately survived. This conservative behavior is critical in high-stakes clinical scenarios, where false-positive predictions can lead to unjustified withdrawal of life-sustaining treatment. Sensitivity, however, remained moderate, particularly for patients who died from non-neurological causes. These patients often show preserved cerebral activity and would likely be rated favorably by clinicians based on EEG alone. Given that the model uses only EEG input, this limitation is not unexpected. Importantly, the model's decision thresholds were tuned to prioritize confidence in poor-outcome predictions, and the resulting calibration favored survivors even though they were the minority class (37 survivors vs. 47 deceased). This reflects a deliberate design choice appropriate for the clinical context.

While k-NN was a natural choice given its non-parametric nature and strong performance under small-sample constraints, the choice of classifier remains an possible area for improvement. Preliminary testing with logistic regression and random forests did not outperform the k-NN baseline and suffered from instability or overfitting, especially in the high-dimensional (320D) latent space (see Section 4.2). However, this does not rule out future gains. Reducing the dimension of the representation space, may stabilize training and enable the use of more expressive classifiers. Furthermore, switching to a trainable classifier would allow the introduction of class-reweighting strategies, which emphasize underrepresented outcomes during training. These methods may help boost sensitivity in future iterations, particularly in clinical settings where false negatives are a key concern.

The modular design of the pipeline, with a stand-alone TS2Vec encoder followed by a k-NN classifier,

enables flexible reuse. Once the representations are extracted, the encoder can be paired with alternative classifiers or used for similarity-based querying in larger archives. This supports downstream use cases such as case-based reasoning or large-scale EEG indexing. However, this architectural choice also comes with trade-offs. Because the encoder is not fine-tuned for the classification task, its representations may be suboptimal for outcome prediction. Integrating the encoder and classifier into a single end-to-end model would allow for joint optimization and potentially improved performance. However, it would sacrifice plug-and-play flexibility, require heavier computation for every new downstream task, and reduce interpretability.

Importantly, the modular setup also avoids potential gradient interference effects observed in hybrid self-supervised learning methods, where multiple (pretext) tasks are optimized jointly [50, Sec. 6]. As highlighted by Weng et al., such interference can degrade representation quality if the losses are not carefully balanced. By contrast, our approach isolates the representation learning phase from classification, ensuring that the contrastive objective is optimized without competing signals. While modest gains might be achievable through end-to-end fine-tuning, the decoupled approach already achieves an AUC of 0.86 on the primary endpoint, and we consider this an acceptable trade-off for a first clinical proof-of-concept.

Direct comparisons with other prognostic pipelines, such as COMET, TS-TCC and CA-TCC, are complicated by differences in modeling objectives and evaluation strategy. In this study, we explicitly constrained the classifier to avoid any false-positive predictions, prioritizing perfect precision and specificity. This naturally reduces sensitivity compared to pipelines that target balanced performance metrics. If COMET or TS-TCC were re-calibrated under the same clinical constraint, their performance would likely converge toward that of the present model. Conversely, relaxing the constraint in our pipeline would increase sensitivity and therefore accuracy, but at the cost of some reduced specificity. The conservative setting adopted here is therefore not only methodologically distinct but clinically motivated.

Interpretability remains a central challenge. While t-SNE visualizations provide some insight into how the encoder organizes data, they do not explain which features of the EEG are driving the model's predictions. Saliency maps, evaluated using both norm- and direction-based metrics, were highly sensitive to design choices and failed to yield consistent or clinically meaningful insights. For example, norm-based maps were dominated by amplitude effects, while directional maps lacked intuitive structure. This reflects a broader limitation of deep learning methods in medical contexts and highlights the need for better attribution tools. Techniques such as integrated gradients or clinically validated benchmarks may improve interpretability in future studies.

Finally, several limitations must be acknowledged. The dataset is small (84 patients), single-center, and focused on a highly specific population: comatose children following cardiac arrest. This constrains the generalizability of the results. Moreover, outcome labels were binarized from the PCPC scale, which may obscure clinically relevant distinctions among survivors. In a supplementary multiclass analysis using the full six-level PCPC scale, the classifier identified deaths more accurately but failed to distinguish among surviving patients. This underscores the intrinsic difficulty of EEG-based multiclass prognostication, especially when distinctions are subtle or clinically ambiguous.

To further characterize this behavior, we computed a cumulative accuracy metric based on whether the predicted PCPC score was at least as severe as the true score ($\hat{y} \geq y_{\text{true}}$). This yielded an overall accuracy of 0.738, indicating that while exact predictions were often incorrect, the model tended to lean toward caution by overestimating outcome severity. Prognosis is further complicated by unmeasured confounders such as sedation level, hypothermia, or ventilator settings—none of which were included in the model. Given these limitations, the model should be viewed as a proof of concept. Additionally, the choice of 20-second EEG epochs, while consistent with prior work, may not be optimal. Different window lengths could capture distinct aspects of EEG dynamics and merit systematic exploration. Any clinical deployment would require external validation, calibration to local practice, and ongoing clinician oversight.

7.1. Future Work

A key limitation of the present study is the single-center source of data. To evaluate generalizability, future work should involve external validation on data from other hospitals. This would help determine the robustness of the encoder to variations in EEG acquisition, artifact removal, and patient populations. Including data from hospitals in different countries may also result in data where patients with PCPC label 5 are included, which are not in our current dataset due to clinical practice in the Netherlands.

A natural next step is to exploit domain-specific structure that the current pipeline does not utilize. First, representing the scalp electrode montage as a graph and training a graph-contrastive encoder could more effectively capture spatial relationships, such as local electrode adjacency and long-range inter-hemispheric connections, than the current flat channel concatenation [50, Sec. 7]. Second, incorporating auxiliary data streams may further enhance performance. Signals such as ECG, along with patient metadata including age, sex, medication and clinical history, could provide complementary information that supports more accurate outcome prediction.

Model design can also be extended in several ways. Currently, the encoder and classifier are trained independently, which limits the ability to fine-tune representations for outcome prediction. Integrating these components into an end-to-end architecture would allow for joint optimization and may improve discriminative power, especially for borderline or ambiguous cases, despite the possible gradient interference effects. Additionally, hybrid training regimes that combine self-supervised and supervised losses could leverage available labels without discarding the benefits of unsupervised pretraining. Incorporating patient-aware sampling, such as positive- or negative-pair selection based on metadata or temporal context, could further improve representation quality. Finally, expanding the label space to include the full PCPC scale or continuous recovery metrics could enable the model to support richer prognostic outputs, though this will require larger labeled datasets and perhaps more granular clinical annotations than the PCPC scale.

Improving interpretability remains a crucial challenge. The saliency methods evaluated in this study were unstable and difficult to relate to clinical concepts. Future work should explore more principled attribution techniques such as integrated gradients (which would require the classifier to be connected to the encoder directly).

An additional avenue for future work is to investigate the relationship between the learned latent representations and conventional qEEG features. Comparing encoder outputs with summary metrics such as amplitude, continuity, and spectral content could yield insights into which clinically established features are implicitly captured by the model. Conversely, if certain qEEG dimensions are underrepresented in the latent space, this could inform targeted architecture changes (like how positive samples are designed) or loss design. This analysis may also provide a useful bridge for interpretability by grounding neural network representations in familiar clinical metrics.

The modular encoder also lends itself to retrieval-based tasks. Latent representations could be used to identify similar patients in large historical EEG archives, enabling case-based reasoning. This is particularly promising given that Erasmus MC alone has already collected over 30,000 EEG recordings, suggesting that multi-centre archives could contain orders of magnitude more. When a new patient undergoes EEG monitoring, their recording can be embedded by the encoder and compared to this archive to retrieve the most similar cases. Such similarity-based retrieval may support clinical decision-making by surfacing relevant precedents with known outcomes. Ultimately, real-world testing in human-in-the-loop environments will be critical to evaluate the safety, usability, and clinical impact of such tools before deployment in routine care.

Finally, clinical integration of contrastive learning models for EEG classification will require careful design beyond algorithmic performance. Threshold calibration could allow clinicians to adjust the models' operating points to match risk tolerance in different settings, such as intensive care or follow-up planning. While this thesis presents a proof of concept, any future clinical application of contrastive learning models will require expert oversight and careful ethical integration, with clear communication of model limitations to support responsible use.

References

[1] Marjolein M. Admiraal et al. “EEG in a four-electrode frontotemporal montage reliably predicts outcome after cardiac arrest”. In: *Resuscitation* 188 (2023), p. 109817. issn: 0300-9572. doi: https://doi.org/10.1016/j.resuscitation.2023.109817. url: https://www.sciencedirect.com/science/article/pii/S0300957223001302.

[2] Edilberto Amorim et al. “Quantitative EEG reactivity and machine learning for prognostication in hypoxic-ischemic brain injury”. In: *Clinical Neurophysiology* 130.10 (2019), pp. 1908–1916. issn: 1388-2457. doi: https://doi.org/10.1016/j.clinph.2019.07.014. url: https://www.sciencedirect.com/science/article/pii/S1388245719311605.

[3] Baburov. https://www.medicalnewstoday.com/articles/325191. [Photograph]. 2009.

[4] Sofia Backman et al. “Reduced EEG montage has a high accuracy in the post cardiac arrest setting”. In: *Clinical Neurophysiology* 131.9 (2020), pp. 2216–2223. issn: 1388-2457. doi: https://doi.org/10.1016/j.clinph.2020.06.021. url: https://www.sciencedirect.com/science/article/pii/S1388245720303990.

[5] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford, UK: Oxford University Press, 1995. isbn: 9780198538646.

[6] Leo Breiman. “Random forests”. In: *Machine Learning* 45.1 (2001). Cited by: 93338; All Open Access, Bronze Open Access, pp. 5–32. doi: 10.1023/A:1010933404324. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-0035478854&doi=10.1023%2fA%3a1010933404324&partnerID=40&md5=4b9f43897146098c0df3a2af232cf2f4.

[7] Christopher J. C. Burges. “A tutorial on support vector machines for pattern recognition”. In: *Data Mining and Knowledge Discovery* 2.2 (1998). Cited by: 14232, pp. 121–167. doi: 10.1023/A:1009715923555. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-27144489164&doi=10.1023%2fA%3a1009715923555&partnerID=40&md5=68510b6561d8c012d7789f9cad38023e.

[8] Luca Cagliero, Silvia Buccafusco, and Francesco Vaccarino. “Contrastive Learning for Multivariate Time Series Classification: an Early Fusion Approach”. In: *17th International Conference on Application of Information and Communication Technologies (AICT)*. IEEE. 2023. doi: 10.1109/AICT59525.2023.10313147. url: https://ieeexplore.ieee.org/document/10313147.

[9] S.D. Caprarola, S.R. Kudchadkar, and M.M. Bembea. “Neurologic Outcomes Following Care in the Pediatric Intensive Care Unit”. In: *Curr Treat Options Peds* 3 (2017), pp. 193–207. doi: https://doi.org/10.1007/s40746-017-0092-x. url: https://link.springer.com/article/10.1007/s40746-017-0092-x.

[10] Ting Chen et al. “A simple framework for contrastive learning of visual representations”. In: vol. PartF168147-3. Cited by: 7126. 2020, pp. 1575–1585. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85105185174&partnerID=40&md5=29b4d3929d35325ea4a9a495e5effd39.

[11] Chris Ding and Xiaofeng He. “K-means clustering via principal component analysis”. In: Cited by: 976. 2004, pp. 225–232. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-14344257496&partnerID=40&md5=09310cc192cf973a0d4a7c2fcbc9e39f.

[12] EITCA Academy. *Challenges in Evaluating the Effectiveness of Unsupervised Learning Algorithms*. Accessed: 2024-02-20. 2024. url: https://eitca.org/artificial-intelligence/eitc-ai-adl-advanced-deep-learning/unsupervised-learning/unsupervised-representation-learning/examination-review-unsupervised-representation-learning/what-are-the-challenges-associated-with-evaluating-the-effectiveness-of-unsupervised-learning-algorithms-and-what-are-some-potential-methods-for-this-evaluation/.

[13] Emadeldeen Eldele et al. “Self-Supervised Contrastive Representation Learning for Semi-Supervised Time-Series Classification”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.12 (Dec. 2023), pp. 15604–15618. issn: 1939-3539. doi: 10.1109/tpami.2023.3308189. url: http://dx.doi.org/10.1109/TPAMI.2023.3308189.

[14] F. W. Fung et al. “Early EEG Features for Outcome Prediction After Cardiac Arrest in Children”. In: *Journal of Clinical Neurophysiology* 36.5 (2019), pp. 349–357. doi: doi:10.1097/wnp.0000000000000591.

[15] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. “Unsupervised Representation Learning by Predicting Image Rotations”. In: *CoRR* abs/1803.07728 (2018). arXiv: 1803.07728. url: http://arxiv.org/abs/1803.07728.

[16] Alexandre Gramfort et al. “MEG and EEG data analysis with MNE-Python”. In: *Frontiers in Neuroscience* Volume 7 - 2013 (2013). issn: 1662-453X. doi: 10.3389/fnins.2013.00267. url: https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2013.00267.

[17] Dan Hendrycks and Kevin Gimpel. *Gaussian Error Linear Units (GELUs)*. 2023. arXiv: 1606.08415 [cs.LG]. url: https://arxiv.org/abs/1606.08415.

[18] L. J. Hirsch et al. “American Clinical Neurophysiology Society’s Standardized Critical Care EEG Terminology: 2021 Version”. In: *Journal of Clinical Neurophysiology* 38.1 (2021), pp. 1–29. doi: 10.1097/WNP.0000000000000806.

[19] Maayke Hunfeld et al. “Prediction of Survival After Pediatric Cardiac Arrest Using Quantitative EEG and Machine Learning Techniques”. In: *Neurology* 103.12 (2024), e210043. doi: 10.1212/WNL.0000000000210043. url: https://www.neurology.org/doi/abs/10.1212/WNL.0000000000210043.

[20] A.K. Jain, M.N. Murty, and P.J. Flynn. “Data clustering: A review”. In: *ACM Computing Surveys* 31.3 (1999). Cited by: 10779; All Open Access, Bronze Open Access, pp. 264–323. doi: 10.1145/331499.331504. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84893405732&doi=10.1145%2f331499.331504&partnerID=40&md5=1b4613e4962a766e432181d239bdd6c0.

[21] Gareth James et al. *An Introduction to Statistical Learning: with Applications in R*. Second. Springer, 2021. isbn: 978-1-0716-1418-1. doi: 10.1007/978-1-0716-1418-1.

[22] Prannay Khosla et al. “Supervised contrastive learning”. In: vol. 2020-December. Cited by: 2892. 2020. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85108420032&partnerID=40&md5=37450af962d9b733bd1e3b79f60c7de8.

[23] Z.J. Koles. “The quantitative extraction and topographic mapping of the abnormal components in the clinical EEG”. In: *Electroencephalography and Clinical Neurophysiology* 79.6 (1991), pp. 440–447. issn: 0013-4694. doi: https://doi.org/10.1016/0013-4694(91)90163-X. url: https://www.sciencedirect.com/science/article/pii/001346949190163X.

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “ImageNet classification with deep convolutional neural networks”. In: *Communications of the ACM* 60.6 (2017). Cited by: 22526; All Open Access, Bronze Open Access, Green Open Access, pp. 84–90. doi: 10.1145/3065386. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85020126914&doi=10.1145%2f3065386&partnerID=40&md5=d920d0868678405baf0e4410804358ec.

[25] Solomon Kullback and Richard A. Leibler. “On Information and Sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86. doi: 10.1214/aoms/1177729694.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep Learning”. In: *arXiv preprint arXiv:1206.5538* (2012). url: https://arxiv.org/abs/1206.5538.

[27] Seungha Lee et al. “Quantitative EEG predicts outcomes in children after cardiac arrest”. In: *Neurology* 92.20 (2019), E2329–E2338. doi: 10.1212/WNL.0000000000007504.

[28] Yutong Li et al. “Balance-energy of resting-state network in obsessive–compulsive disorder”. In: *Journal of Affective Disorders* 341 (2023), pp. 319–327. doi: 10.1016/j.jad.2023.08.032.

[29] Ziyu Liu et al. “Self-Supervised Contrastive Learning for Medical Time Series: A Systematic Review”. In: *Sensors* 23.9 (2023), p. 4221. doi: 10.3390/s23094221. url: https://doi.org/10.3390/s23094221.

[30] Laurens van der Maaten and Geoffrey Hinton. “Visualizing Data using t-SNE”. In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605.

[31] Qianwen Meng et al. “Unsupervised Representation Learning for Time Series: A Review”. In: *Journal of LaTeX Class Files* 14.8 (Aug. 2021), pp. 1–25. arXiv: 2308.01578 [cs.LG]. url: https://arxiv.org/abs/2308.01578.

[32] Mohammad Amin Morid, Olivia R. Liu Sheng, and Joseph Dunbar. “Time Series Prediction Using Deep Learning Methods in Healthcare”. In: *ACM Transactions on Management Information Systems* 14.1 (2023). doi: 10.1145/3531326.

[33] Alexey Natekin and Alois Knoll. “Gradient boosting machines, a tutorial”. In: *Frontiers in Neurorobotics* 7.DEC (2013). Cited by: 2194; All Open Access, Gold Open Access, Green Open Access. doi: 10.3389/fnbot.2013.00021. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84892667860&doi=10.3389%2ffnbot.2013.00021&partnerID=40&md5=039319aa46825b7e503022916d16725b.

[34] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[35] Kasuni Perera et al. “EEG Patterns and Outcomes After Hypoxic Brain Injury: A Systematic Review and Meta-analysis”. In: *Neurocritical Care* 36.1 (2022), pp. 292–301. doi: 10.1007/s12028-021-01322-0.

[36] Stanley D.T. Pham et al. “Outcome Prediction of Postanoxic Coma: A Comparison of Automated Electroencephalography Analysis Methods”. In: *Neurocritical Care* 37.SUPPL 2 (2022), pp. 248–258. doi: 10.1007/s12028-022-01449-8. url: https://link.springer.com/article/10.1007/s12028-022-01449-8.

[37] Emanuele Roberti, Giorgio Chiarini, Nicola Latronico, et al. “Electroencephalographic monitoring of brain activity during cardiac arrest: a narrative review”. In: *ICMx* 11.1 (2023), p. 4. doi: 10.1186/s40635-022-00489-w.

[38] S.T. Roweis and L.K. Saul. “Nonlinear dimensionality reduction by locally linear embedding”. In: *Science* 290.5500 (2000). Cited by: 13102, pp. 2323–2326. doi: 10.1126/science.290.5500.2323. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-0034704222&doi=10.1126%2fscience.290.5500.2323&partnerID=40&md5=a2ab45f99b0ca02b5136b82b0b9419b8.

[39] Luc Rubinger et al. “Machine learning and artificial intelligence in research and healthcare”. In: *Injury* 54 (2023). AOTrauma Europe Supplement: Clinical Research: Lessons Learned-Looking Ahead, S69–S73. issn: 0020-1383. doi: https://doi.org/10.1016/j.injury.2022.01.046. url: https://www.sciencedirect.com/science/article/pii/S0020138322000766.

[40] Claudio Sandroni et al. “Prediction of good neurological outcome in comatose survivors of cardiac arrest: a systematic review”. In: *Intensive Care Medicine* 48.4 (2022), pp. 389–413. doi: 10.1007/s00134-022-06618-z.

[41] Claudio Sandroni et al. “Prediction of poor neurological outcome in comatose survivors of cardiac arrest: a systematic review”. In: *Intensive Care Medicine* 46.10 (2020), pp. 1803–1851. doi: 10.1007/s00134-020-06198-w.

[42] Jürgen Schmidhuber. “Deep Learning in neural networks: An overview”. In: *Neural Networks* 61 (2015). Cited by: 13948; All Open Access, Green Open Access, pp. 85–117. doi: 10.1016/j.neunet.2014.09.003. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84910651844&doi=10.1016%2fj.neunet.2014.09.003&partnerID=40&md5=1e380e40a7a616540c705ef8a63ce456.

[43] Pierre Sermanet et al. “Time-Contrastive Networks: Self-Supervised Learning from Video”. In: Cited by: 521; All Open Access, Green Open Access. 2018, pp. 1134–1141. doi: 10.1109/ICRA.2018.8462891. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85060855803&doi=10.1109%2fICRA.2018.8462891&partnerID=40&md5=3b14dcc083de189081962b24d8245bb6.

[44] Paul M. Shore. “Prediction of outcome after pediatric cardiac arrest: No crystal ball yet”. In: *Pediatric Critical Care Medicine* 8.1 (2007), pp. 72–73. doi: 10.1097/01.pcc.0000256613.47888.d9.

[45] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2014. url: https://arxiv.org/abs/1312.6034.

[46] *The Medicine Study Electrophysiology*. 2020. url: https://pixabay.com/photos/the-medicine-study-electrophysiology-4764731/.

[47] Michael Tschannen et al. *On Mutual Information Maximization for Representation Learning*. 2020. arXiv: 1907.13625 [cs.LG]. url: https://arxiv.org/abs/1907.13625.

[48] Marit Verboom. "Electroencephalography Monitoring in the Critically Ill: Towards a More Efficient and Effective Monitoring Strategy". MA thesis. Leiden University, Delft University of Technology, Erasmus University Rotterdam, 2023. url: https://repository.tudelft.nl/record/uuid:8881fc18-4001-424e-99ee-f6b70b39c2d4.

[49] Yihe Wang et al. *Contrast Everything: A Hierarchical Contrastive Framework for Medical Time-Series*. 2023. arXiv: 2310.14017 [cs.LG]. url: https://arxiv.org/abs/2310.14017.

[50] Weining Weng et al. "Self-supervised Learning for Electroencephalogram: A Systematic Survey". In: *ACM Computing Surveys* (May 2025). In press, accepted May 2025. doi: 10.1145/3736574.

[51] Thomas Wolf et al. "Transformers: State-of-the-Art Natural Language Processing". In: Cited by: 7248. 2020, pp. 38–45. url: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85173485086&partnerID=40&md5=25e3c4a576f5a1f1589756e1dbad3074.

[52] Xue Yang, Xuejun Qi, and Xiaobo Zhou. "Deep Learning Technologies for Time Series Anomaly Detection in Healthcare: A Review". In: *IEEE Access* 11 (2023), pp. 117788–117799. doi: 10.1109/ACCESS.2023.3325896.

[53] Zihan Yue et al. "TS2Vec: Towards Universal Representation of Time Series". In: *arXiv preprint arXiv:2106.10466* (Feb. 2022). url: https://arxiv.org/abs/2106.10466.

[54] Kexin Zhang et al. "Self-Supervised Learning for Time Series Analysis: Taxonomy, Progress, and Prospects". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46.10 (2024), pp. 6775–6794. doi: 10.1109/TPAMI.2024.3387317.

A

Mathematical Description of the Dilated Convolutional Encoder in TS2Vec

The encoder used in this project is a dilated convolutional neural network designed to extract structured representations from multivariate time series. This appendix provides a detailed mathematical description of the ten residual blocks, including the convolutions, dilation, residual connections, and nonlinear activations. For completeness, we also repeat the inner working of the first input projection layer.

Input Projection

Each 20-second EEG recording is represented as a multivariate time series $\boldsymbol{x}_i \in \mathbb{R}^{T \times C}$, where T denotes the number of time points and $C = 4$ is the number of EEG channels. For each time index $t \in \{1, \ldots, T\}$, the row vector $\boldsymbol{x}_{i,t} \in \mathbb{R}^C$ corresponds to the simultaneous measurements of all channels at time t for sample i. Before any temporal processing is applied, the model maps each of these C-dimensional input vectors into a higher-dimensional latent space $\mathbb{R}^D$ using a shared affine transformation. This step is performed independently at each time point and is referred to as the input projection layer:

$$\boldsymbol{u}_{i,t} = \boldsymbol{W}_{\text{in}} \boldsymbol{x}_{i,t} + \boldsymbol{b}_{\text{in}} \in \mathbb{R}^D, \quad \text{for } t = 1, \ldots, T, \tag{A.1}$$

where $\boldsymbol{W}_{\text{in}} \in \mathbb{R}^{D \times C}$ and $\boldsymbol{b}_{\text{in}} \in \mathbb{R}^D$ are learnable parameters of the model. In this work, we use $D = 64$. The result is a transformed sequence $\boldsymbol{U}_i = (\boldsymbol{u}_{i,1}, \ldots, \boldsymbol{u}_{i,T}) \in \mathbb{R}^{T \times D}$, which retains the temporal structure of the original EEG segment but embeds each observation into a richer latent representation space.

This transformation enables the model to better capture interactions between EEG channels and to operate in a feature space more suitable for deep representation learning. Conceptually, it is analogous to expanding the feature space in classical machine learning, where such embeddings allow for more expressive modeling and improved separation of task-relevant patterns.

After the input projection layer, random timestamps in the latent sequence $\boldsymbol{U}_i$ are masked, the process and motivation are described in Section 3.3.3.

Residual Blocks

The core of the encoder consists of a deep stack of ten residual blocks, each designed to extract temporal patterns from the latent sequence $\boldsymbol{U}_i \in \mathbb{R}^{T \times D}$ produced by the input projection layer. Each block applies two successive 1D convolutions, each followed by a nonlinearity function, and adds a shortcut connection known as a residual branch. These blocks are applied sequentially, so that the output of block $\ell - 1$ becomes the input to block ℓ.

Let $\boldsymbol{A}_i^{(\ell)} \in \mathbb{R}^{T \times D_\ell}$ denote the output of the ℓ-th residual block, such that

$$\boldsymbol{A}_i^{(\ell)} = (\boldsymbol{a}_{i,1}^{(\ell)}, \ldots, \boldsymbol{a}_{i,T}^{(\ell)}), \quad \text{with } \boldsymbol{a}_{i,t}^{(\ell)} \in \mathbb{R}^{D_\ell}$$

, and define $\boldsymbol{A}_i^{(0)} := \boldsymbol{U}_i$ as the input to the first block. Each block computes a transformation of the form:

$$\boldsymbol{A}_i^{(\ell)} = \boldsymbol{F}^{(\ell)}\left(\boldsymbol{A}_i^{(\ell-1)}\right) + \boldsymbol{R}_i^{(\ell)}, \quad \ell = 1, \ldots, 10, \tag{A.2}$$

where $\boldsymbol{F}^{(\ell)}$ denotes the main convolutional path of the block, and $\boldsymbol{R}_i^{(\ell)}$ is the residual branch. The function $\boldsymbol{F}^{(\ell)}$ is defined as:

$$\boldsymbol{F}^{(\ell)}(\cdot) := \mathrm{Conv}_2^{(\ell)}\left(\mathrm{GELU}\left(\mathrm{Conv}_1^{(\ell)}\left(\mathrm{GELU}(\cdot)\right)\right)\right), \tag{A.3}$$

where $\mathrm{Conv}_1^{(\ell)}$ and $\mathrm{Conv}_2^{(\ell)}$ denote 1D convolutional operators with dilation factor $d_\ell = 2^{\ell-1}$, kernel size $K = 3$, and symmetric zero-padding to preserve the temporal length T. Each convolution is applied along the time axis and maps input vectors of dimension $D_{\ell-1}$ to output vectors of dimension D_ℓ at every time step.

The purpose of dilation is to increase the receptive field of each convolutional layer exponentially with depth, so that each output vector $\boldsymbol{a}_{i,t}^{(\ell)}$ can incorporate temporal information from a wide neighborhood around time t without increasing the number of parameters or layers, this is visualized in Figure A.1.

The residual branch $\boldsymbol{R}_i^{(\ell)}$ allows each block to refine its input rather than replace it entirely. Specifically, the block learns a perturbation $\boldsymbol{F}^{(\ell)}(\boldsymbol{A}_i^{(\ell-1)})$ to be added to the identity mapping. This structure stabilizes training by preserving information across layers and enables the model to incrementally adjust features, rather than having to relearn them from scratch at each depth. When the input and output dimensions match ($D_{\ell-1} = D_\ell$), the residual connection is the identity:

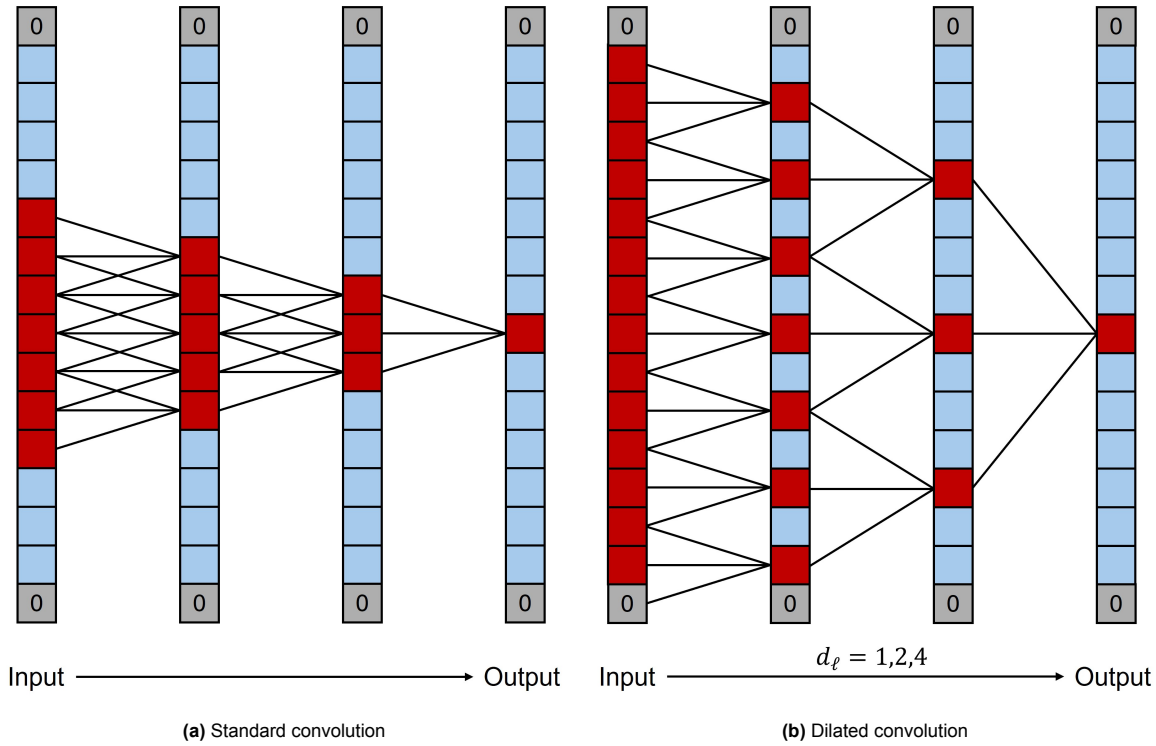
$$\boldsymbol{R}_i^{(\ell)} = \boldsymbol{A}_i^{(\ell-1)}, \quad \text{for } \ell = 1, \ldots, 9. \tag{A.4}$$

In the final block ($\ell = 10$), the feature dimension increases from $D_9 = 64$ to $D_{10} = 320$, and the residual path must be projected accordingly. This is achieved by applying a 1×1 convolution (i.e., a learnable linear transformation at each time step):

$$\boldsymbol{R}_i^{(10)} = \left(\boldsymbol{W}_{\text{res}}\boldsymbol{a}_{i,t}^{(9)} + \boldsymbol{b}_{\text{res}}\right)_{t=1}^{T}, \tag{A.5}$$

where $\boldsymbol{W}_{\text{res}} \in \mathbb{R}^{320 \times 64}$ and $\boldsymbol{b}_{\text{res}} \in \mathbb{R}^{320}$ are shared across time. This ensures that the element-wise addition in (A.2) is well-defined.

The use of residual connections is a critical architectural feature for training deep neural networks. They improve gradient flow during optimization and allow each block to refine, rather than entirely replace, the representation learned in earlier layers. In effect, the network learns residual functions of the form $\boldsymbol{F}^{(\ell)} \approx \boldsymbol{A}_i^{(\ell)} - \boldsymbol{A}_i^{(\ell-1)}$, enabling more stable and efficient convergence.



(a) Standard convolution **(b)** Dilated convolution

Figure A.1: Comparison between standard and dilated 1D convolutions. In red the receptive field of one timestamp of the output, symmetric zero-padding has been applied. Dilated convolutions increase the receptive field exponentially with depth by skipping input positions, using a fixed kernel size ($K = 3$) and no additional parameters.

Each residual block in the encoder applies one-dimensional convolutional operations along the temporal axis of the latent sequence. At a high level, a 1D convolution combines information from a fixed-size window of past and future time points to compute a new representation at each time step. This can be interpreted as a localized linear transformation that shares weights across time.

Formally, consider an input sequence $\boldsymbol{A}^{(\ell-1)}$ (for this section and the nonlinearity (GELU) section we omit the i of $\boldsymbol{A}_i^{(\ell-1)}$ that denotes the instance to improve reading clarity) with $\boldsymbol{A}^{(\ell-1)} = (\boldsymbol{a}_1^{(\ell-1)}, \ldots, \boldsymbol{a}_T^{(\ell-1)}) \in \mathbb{R}^{T \times D_{\ell-1}}$, where each vector $\boldsymbol{a}_t^{(\ell-1)} \in \mathbb{R}^{D_{\ell-1}}$ represents the set of $D_{\ell-1}$ input features at time t in layer $\ell - 1$. The output of the convolutional layer at time t in block ℓ is a vector $\boldsymbol{a}_t^{(\ell)} \in \mathbb{R}^{D_\ell}$, where each component $z_{t,k}^{(\ell)} \in \mathbb{R}$ (for $k = 1, \ldots, D_\ell$) corresponds to one output channel. Each output channel is computed independently using its own convolutional filter and corresponds to a separate feature extractor. The value of channel k at time t is given by:

$$z_{t,k}^{(\ell)} = \sum_{\tau=0}^{K-1} \boldsymbol{w}_\tau^{(\ell,k)} \cdot \boldsymbol{a}_{t+d_\ell \cdot (\tau - \lfloor K/2 \rfloor)}^{(\ell-1)} + b^{(\ell,k)}, \tag{A.6}$$

where:

- K is the kernel size (we use $K = 3$),
- $d_\ell \in \mathbb{N}$ is the dilation factor for layer ℓ, defined as $d_\ell = 2^{\ell-1}$,
- $\boldsymbol{w}_\tau^{(\ell,k)} \in \mathbb{R}^{D_{\ell-1}}$ are the learnable weights of the convolutional filter for offset τ and output channel k,
- $b^{(\ell,k)} \in \mathbb{R}$ is a learnable bias term,
- $\boldsymbol{a}_{t+d_\ell \cdot (\tau - \lfloor K/2 \rfloor)}$ is the input at an earlier timestep, accessed with dilation rate d_ℓ.

The dilation factor d_ℓ introduces gaps between the positions included in the convolutional window. This increases the receptive field, the number of input time points that influence a given output, without increasing the number of learnable parameters or the depth of the network. In particular, using an exponentially increasing dilation schedule $d_\ell = 2^{\ell-1}$ allows the network to capture long-range temporal dependencies in logarithmic depth.

To ensure that the output sequence maintains the same temporal length as the input (T), symmetric zero-padding is applied to the input before each convolution. That is, the sequence $\boldsymbol{A}^{(\ell-1)}$ is extended with zeros on both sides (in the time dimension) so that all T output positions are valid and well-defined, this is also visualized in Figure A.1. This is necessary for positions where $t + d_\ell \cdot (\tau - \lfloor K/2 \rfloor) \leq 0$ or $t + d_\ell \cdot (\tau - \lfloor K/2 \rfloor) \geq T$

Figure A.1 contrasts standard convolutions (with adjacent input positions) to dilated convolutions, which skip over input elements according to a predefined dilation rate. This architectural choice allows the model to extract both short- and long-term temporal features efficiently using only shallow layers and compact kernels.

Nonlinearity (GELU)

Each pre-activation value $z_{t,k}^{(\ell)}$ computed by the convolutional layer is passed through a nonlinear activation function. In this encoder, we use the Gaussian Error Linear Unit (GELU), defined as:

$$\mathrm{GELU}(z) = z \cdot \Phi(z), \tag{A.7}$$

where $\Phi(z)$ denotes the cumulative distribution function of the standard normal distribution.

Unlike the more commonly used Rectified Linear Unit (ReLU), which sets all negative inputs to zero, GELU retains small negative values and applies a smooth, differentiable function, as shown in Figure A.2. This makes GELU particularly suitable for deep networks, as it improves gradient flow and avoids overly sharp nonlinearities that can hinder optimization. The smoothness of GELU encourages stable learning and has been shown to outperform ReLU in many settings [17].

In practice, GELU is applied elementwise to the full matrix of pre-activations at layer ℓ, denoted by $\boldsymbol{Z}^{(\ell)} = (z_{t,k}^{(\ell)}) \in \mathbb{R}^{T \times D_\ell}$. The resulting activation matrix is given by:

$$\boldsymbol{A}^{(\ell)} = \mathrm{GELU}(\boldsymbol{Z}^{(\ell)}) \in \mathbb{R}^{T \times D_\ell}, \tag{A.8}$$

where the GELU function is applied independently to each scalar entry of $\boldsymbol{Z}^{(\ell)}$. This yields a new latent representation at layer ℓ, which is then passed to the next convolutional block.

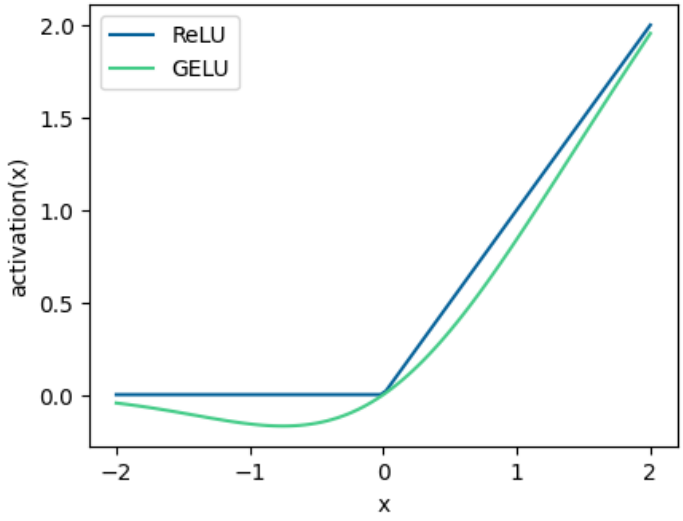


Figure A.2: Comparison of the GELU and ReLU activation functions. GELU transitions smoothly and retains small negative inputs, which improves gradient flow in deep networks.

Output and Pooling

After passing through the final residual block, each EEG segment yields a latent feature sequence

$$\boldsymbol{H}_i = (\boldsymbol{h}_{i,1}, \ldots, \boldsymbol{h}_{i,T}) \in \mathbb{R}^{T \times D_{10}},$$

where $D_{10} = 320$ denotes the output feature dimension of the final layer, and $\boldsymbol{h}_{i,t} \in \mathbb{R}^{320}$ is the latent feature vector corresponding to time step t for sample i.

To obtain a fixed-length representation for downstream classification, a temporal max pooling operation is applied across the entire time axis. Max pooling is used instead of average pooling, because it captures high-magnitude feature activations that may indicate clinically important patterns. Formally, the pooled representation $\boldsymbol{r}_i \in \mathbb{R}^{320}$ is defined as:

$$\boldsymbol{r}_i[k] = \max_{1 \le t \le T} \boldsymbol{h}_{i,t}[k], \quad \text{for } k = 1, \ldots, 320, \tag{A.9}$$

where $\boldsymbol{r}_i[k]$ denotes the k-th feature of the vector $\boldsymbol{r}_i$ and $\boldsymbol{h}_{i,t}[k]$ denotes the k-th feature of the vector $\boldsymbol{h}_{i,t}$. That is, max pooling selects the maximum value across all time steps for each feature channel independently.

This operation reduces the temporal dimension and produces a fixed-size embedding $\boldsymbol{r}_i \in \mathbb{R}^{320}$, regardless of the original sequence length T. The resulting vector summarizes the most salient temporal features present in the EEG segment and serves as input to the downstream classifier.

Training Procedure

The encoder architecture described in this appendix is fully differentiable, meaning its parameters can be optimized using gradient-based methods. The training objective is to minimize the hierarchical contrastive loss, denoted $\mathcal{L}_{\text{hier}}$ (see Section 3.3.2), which encourages the encoder to produce similar output representations for augmented views of the same EEG segment while mapping dissimilar samples farther apart in the latent space.

Let Θ denote the full set of trainable parameters of the encoder. This includes the weights and biases of the input projection layer, all convolutional filters in the residual blocks, and any projection layers used in the residual connections. Since all operations in the encoder (affine projections, dilated convolutions, GELU activations, and max pooling) are differentiable, the loss gradient $\nabla_\Theta \mathcal{L}_{\text{hier}}$ can be computed exactly using the chain rule via backpropagation.

The parameters are updated using the Adam optimizer, a commonly used variant of stochastic gradient descent (SGD) that combines momentum and adaptive per-parameter learning rates. Let $\eta > 0$ denote the learning rate, in this project, we set $\eta = 10^{-4}$. The update rule at each iteration is:

$$\Theta \leftarrow \Theta - \eta \cdot \nabla_\Theta L_{\text{hier}}.$$

Training is performed on the training set only, which consists of approximately 80% of the dataset, or $n_{\text{train}} \approx 0.8 \cdot 6669 = 5335$ EEG segments. These are grouped into mini-batches of size $B = 8$, resulting in roughly

$$\left\lfloor \frac{n_{\text{train}}}{B} \right\rfloor = \left\lfloor \frac{5335}{8} \right\rfloor \approx 666 \quad \text{parameter updates per epoch.}$$

Each training epoch proceeds as follows. A mini-batch $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_B\}$ is sampled from the training set. For each segment $\boldsymbol{x}_i$, an augmented version $\boldsymbol{x}_i'$ is generated using timestamp masking and random cropping (see Section 3.3.3). Both the original and augmented segments are passed through the encoder, yielding representations $\boldsymbol{r}_i = f(\boldsymbol{x}_i; \Theta) \in \mathbb{R}^{320}$ and $\boldsymbol{r}_i' = f(\boldsymbol{x}_i'; \Theta) \in \mathbb{R}^{320}$. The hierarchical contrastive loss is then evaluated on the set of $2B$ vectors.

Algorithm 2 Training of the TS2Vec Encoder

Require: Training set $\mathcal{D}_{\text{train}} = \{\boldsymbol{x}_i\}_{i=1}^{n_{\text{train}}}$, encoder $f(\cdot;\Theta)$, learning rate $\eta = 10^{-4}$, number of epochs $E = 6$, batch size $B = 8$
Ensure: Trained encoder parameters Θ

for epoch $\leftarrow 1$ to E **do**
  for each mini-batch $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_B\} \subset \mathcal{D}_{\text{train}}$ **do**
    Augment (crop) inputs: $\boldsymbol{x}'_i \leftarrow \text{Crop}(\boldsymbol{x}_i)$ for $i = 1, \ldots, B$ (see Section 3.3.3)
    Encode: $\boldsymbol{r}_i \leftarrow f(\boldsymbol{x}_i;\Theta), \quad \boldsymbol{r}'_i \leftarrow f(\boldsymbol{x}'_i;\Theta)$
    Compute contrastive loss:

$$\mathcal{L}_{\text{hier}} \leftarrow \mathcal{L}_{\text{Hier}}(\{\boldsymbol{r}_i\}, \{\boldsymbol{r}'_i\}) \quad \text{(see Algorithm 1)}$$

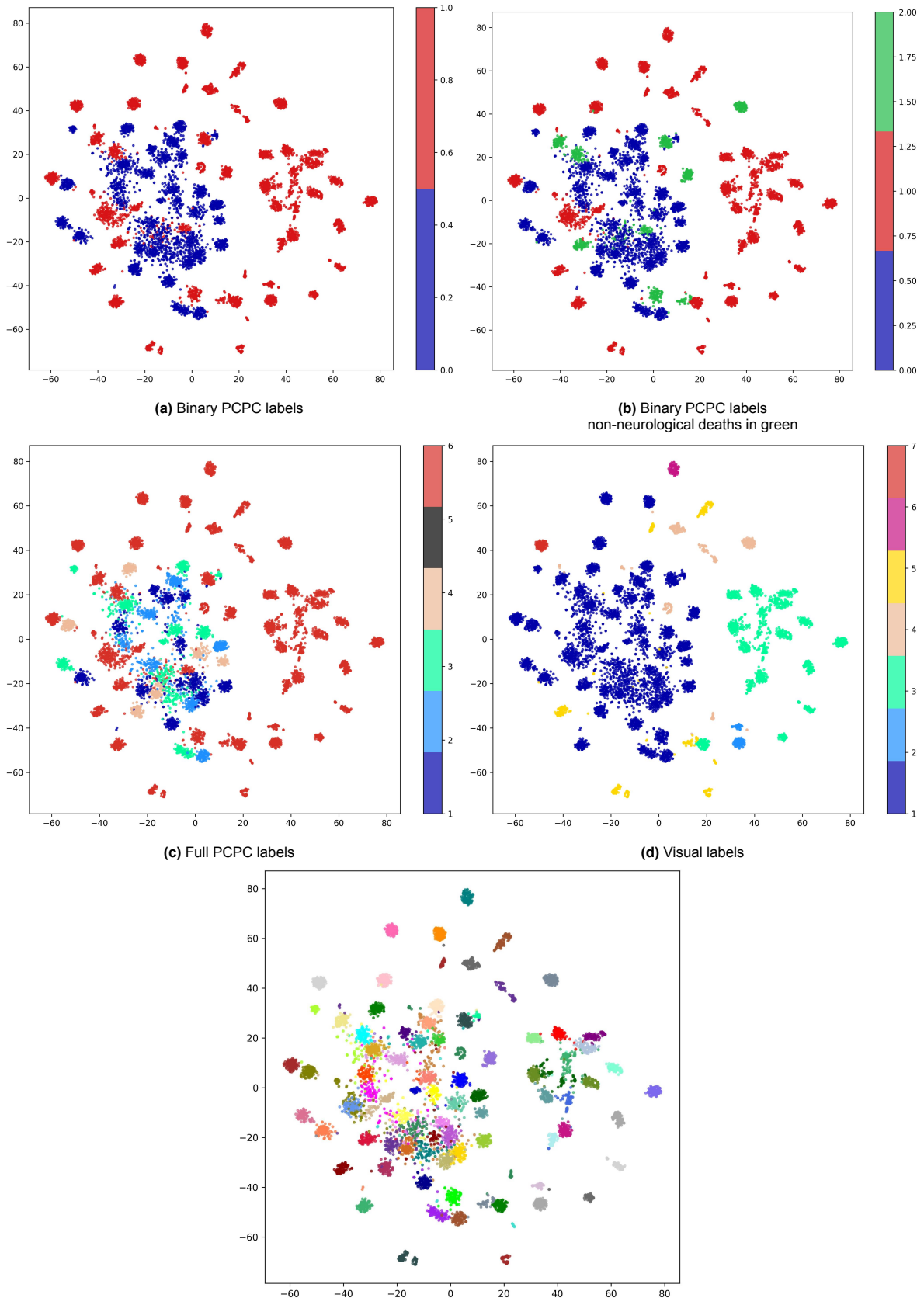
    Compute gradient: $g \leftarrow \nabla_{\Theta} \mathcal{L}_{\text{hier}}$
    Update parameters: $\Theta \leftarrow \Theta - \eta \cdot g$
  end for
end for
return Θ

The encoder is trained for $E = 6$ full epochs over the training set. All parameters in Θ are updated jointly to minimize the loss. The result is a trained encoder capable of mapping raw EEG segments into fixed-length representations that are stable, discriminative, and useful for downstream tasks such as outcome classification.

B

t-SNE Visualizations

The two-dimensional t-SNE embeddings of the representation space for all five cross-validation folds. Fold 1 (also shown in Section 5.1.1) is included again here for completeness. Each point corresponds to a single EEG epoch and is colored by the associated labels for that patient. The PCPC labels are the PCPC score after 12 months.



(a) Binary PCPC labels

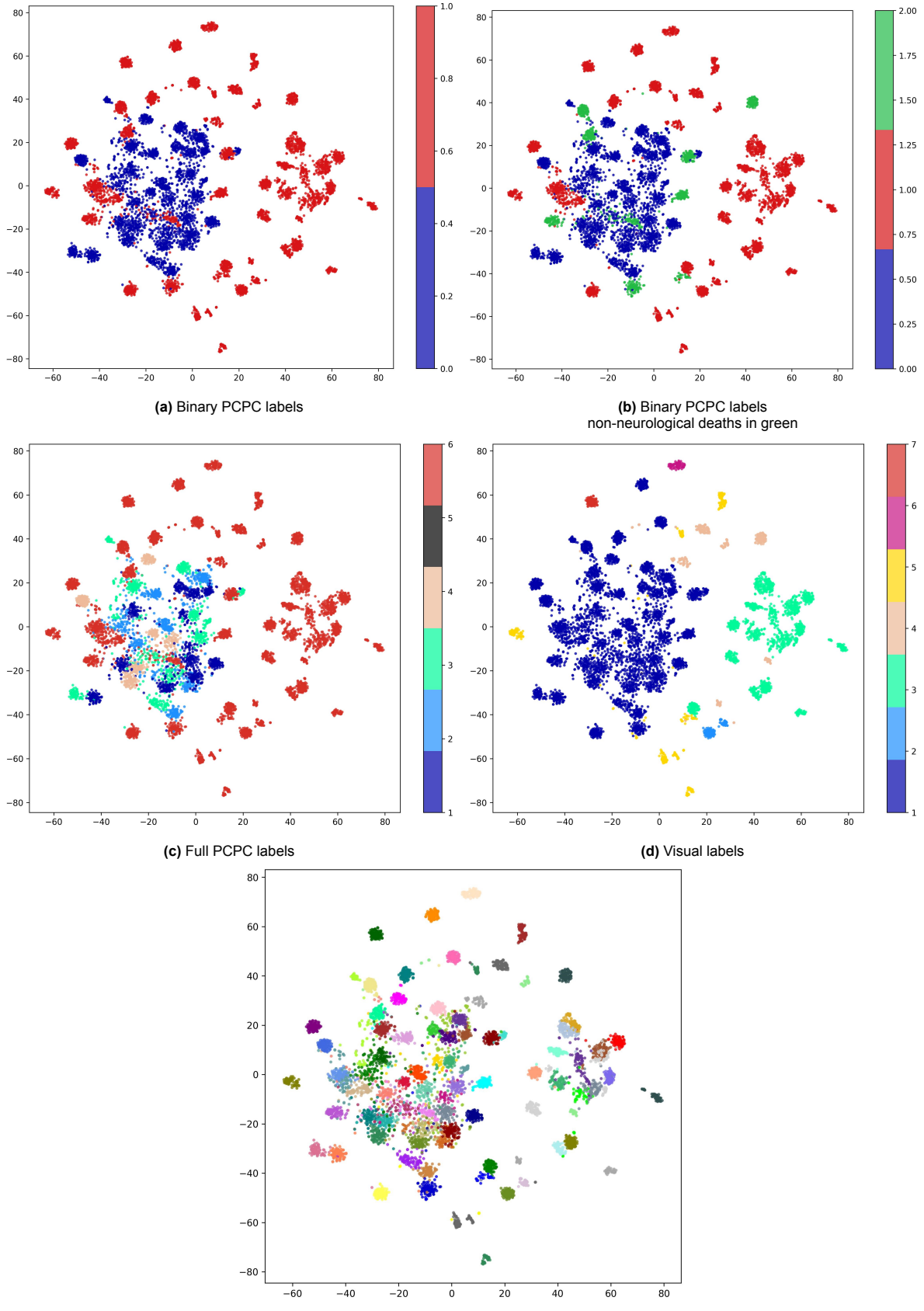
(b) Binary PCPC labels
non-neurological deaths in green

(c) Full PCPC labels

(d) Visual labels

(e) Patient number

Figure B.1: Two-dimensional t-SNE embeddings of encoder representations, fold 0.



(a) Binary PCPC labels

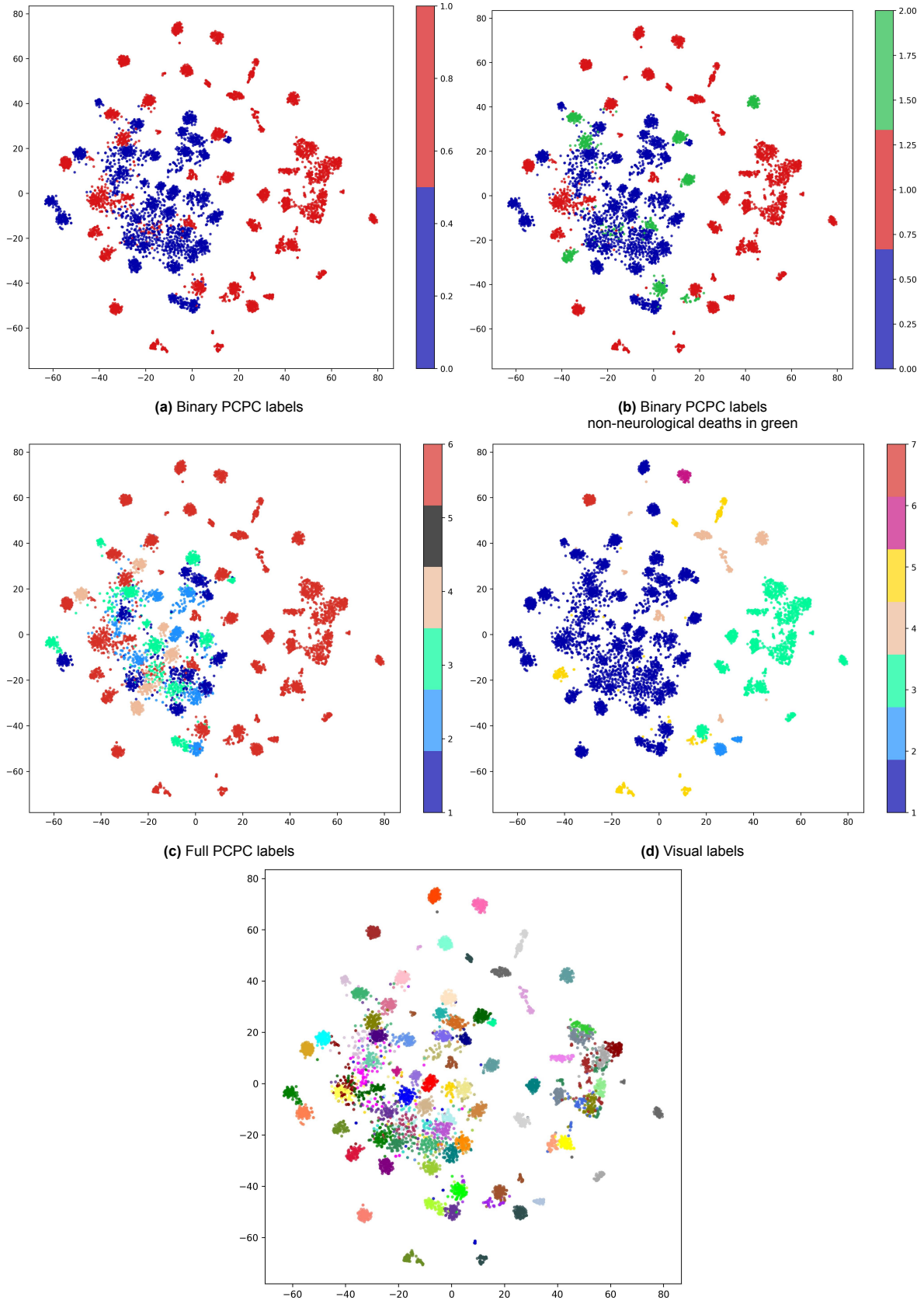
(b) Binary PCPC labels
non-neurological deaths in green

(c) Full PCPC labels

(d) Visual labels

(e) Patient number

Figure B.2: Two-dimensional t-SNE embeddings of encoder representations, fold 1.



(a) Binary PCPC labels

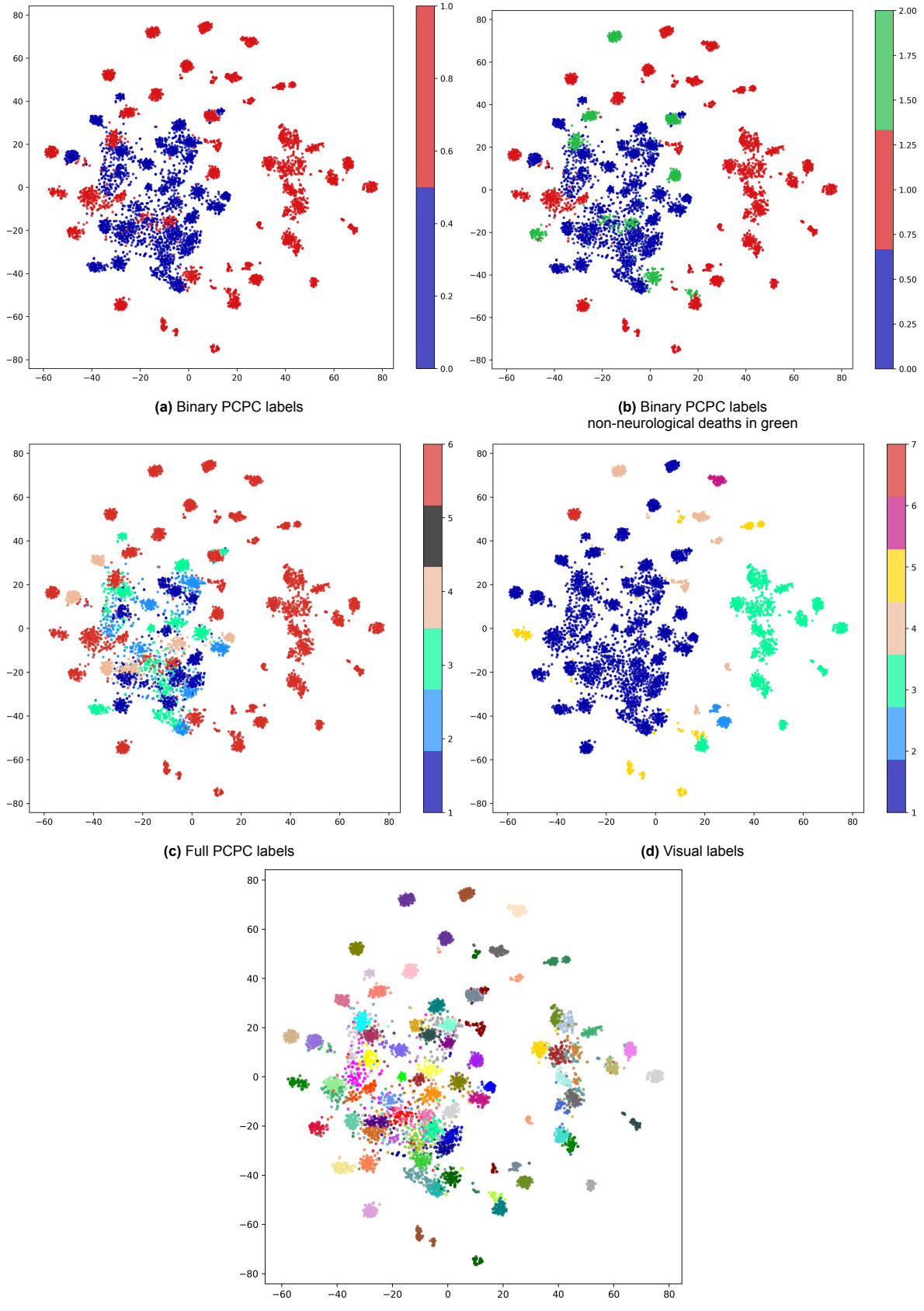
(b) Binary PCPC labels
non-neurological deaths in green

(c) Full PCPC labels

(d) Visual labels

(e) Patient number

Figure B.3: Two-dimensional t-SNE embeddings of encoder representations, fold 2.



(a) Binary PCPC labels

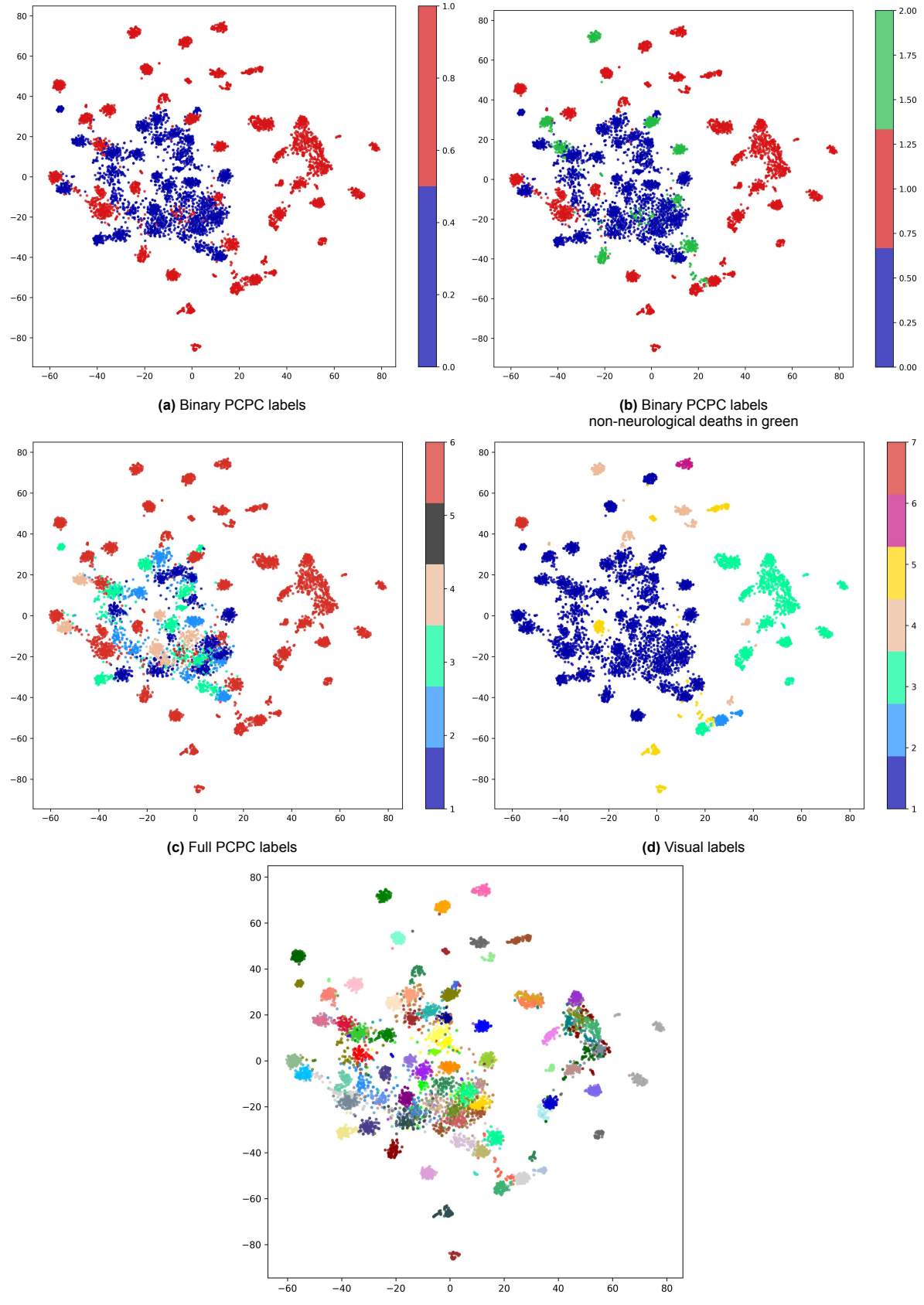
(b) Binary PCPC labels
non-neurological deaths in green

(c) Full PCPC labels

(d) Visual labels

(e) Patient number

Figure B.4: Two-dimensional t-SNE embeddings of encoder representations, fold 3.



(a) Binary PCPC labels

(b) Binary PCPC labels
non-neurological deaths in green

(c) Full PCPC labels

(d) Visual labels

(e) Patient number

Figure B.5: Two-dimensional t-SNE embeddings of encoder representations, fold 4.

C

Saliency Maps

This appendix provides supplementary examples of the saliency maps described in Section 5.1.2. We include both magnitude-based maps and directional maps of epochs of patients with PCPC scores 1, 2, 3, 4 and 6 after 12 months. The epoch belonging to the patient with PCPC score 3 is the same as the epoch shown in Section 5.1.2.

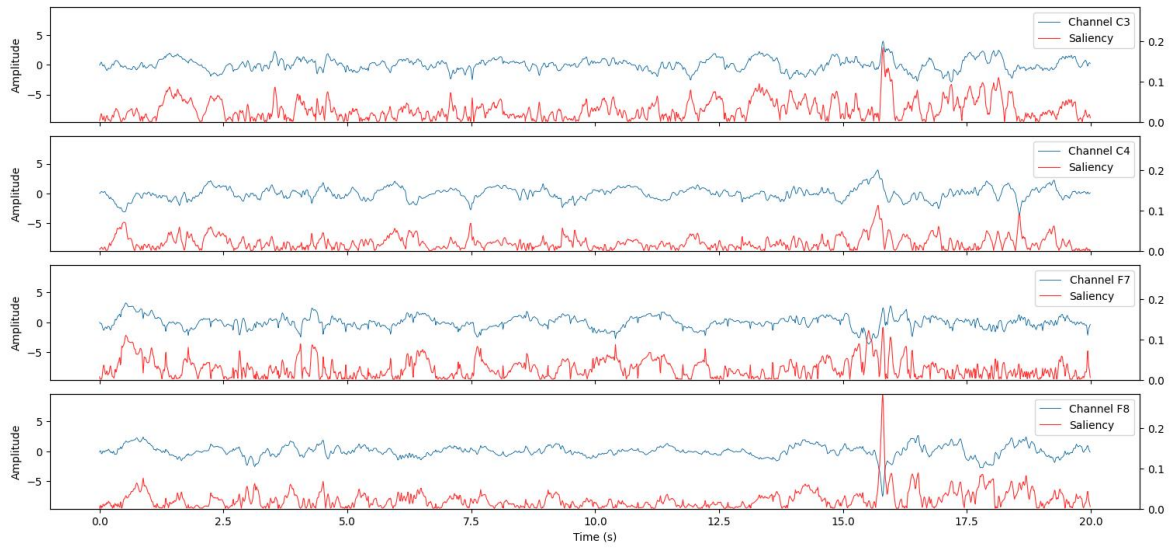


Figure C.1: Magnitude-based saliency map for a 20-second EEG segment from a patient with PCPC score 1 after 12 months.

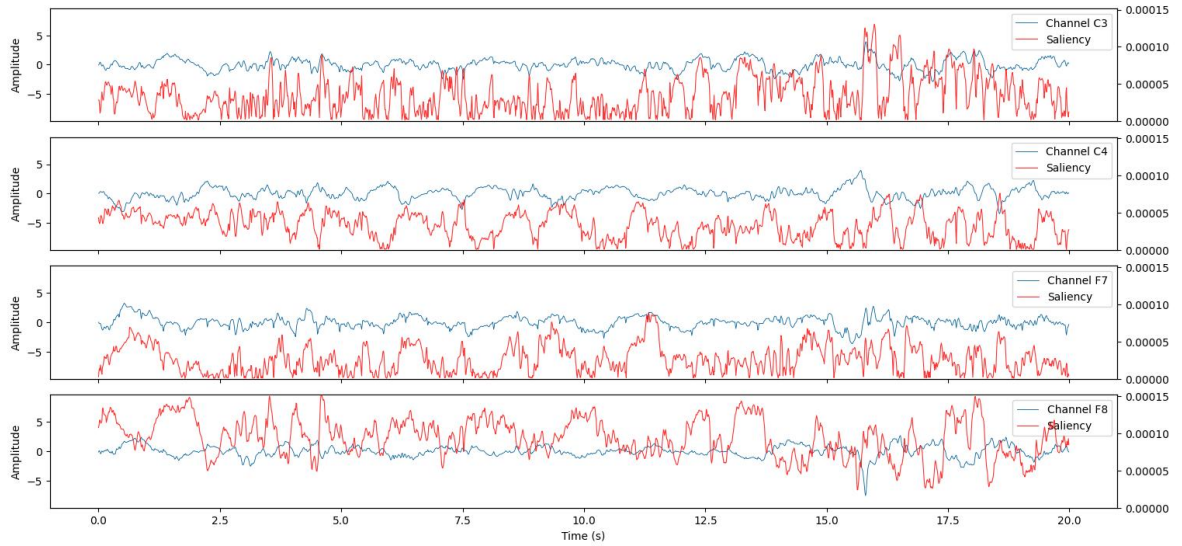


Figure C.2: Directional-based saliency map for a 20-second EEG segment from a patient with PCPC score 1 after 12 months.

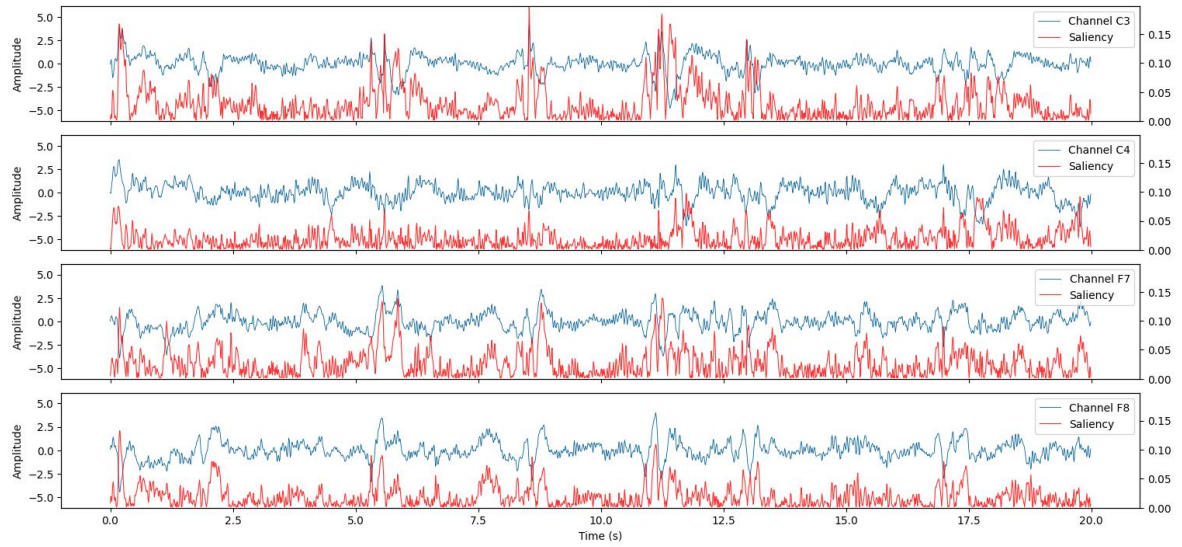


Figure C.3: Magnitude-based saliency map for a 20-second EEG segment from a patient with PCPC score 2 after 12 months.

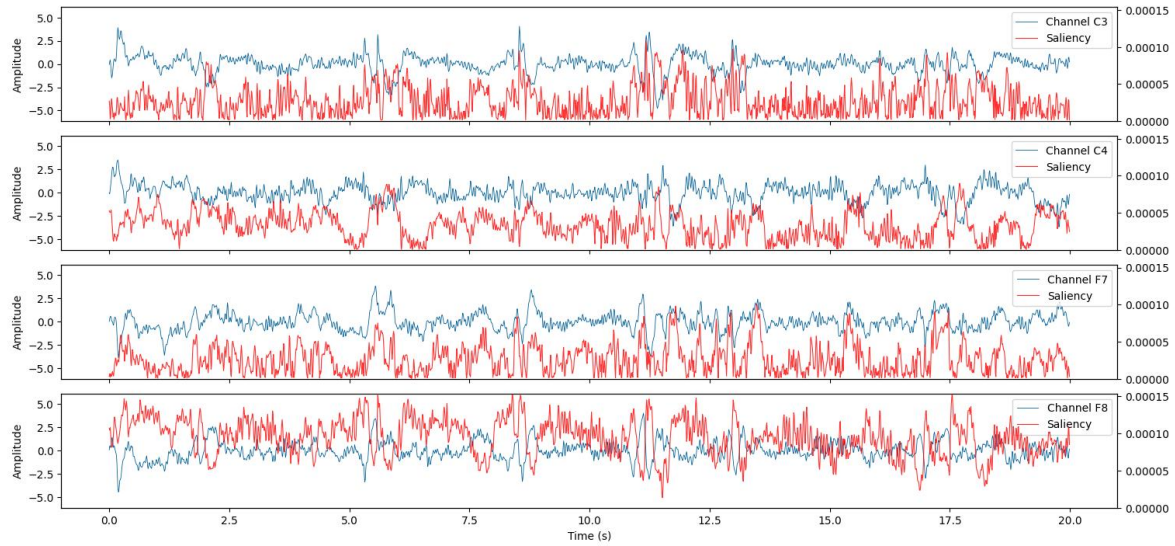


Figure C.4: Directional-based saliency map for a 20-second EEG segment from a patient with PCPC score 2 after 12 months.

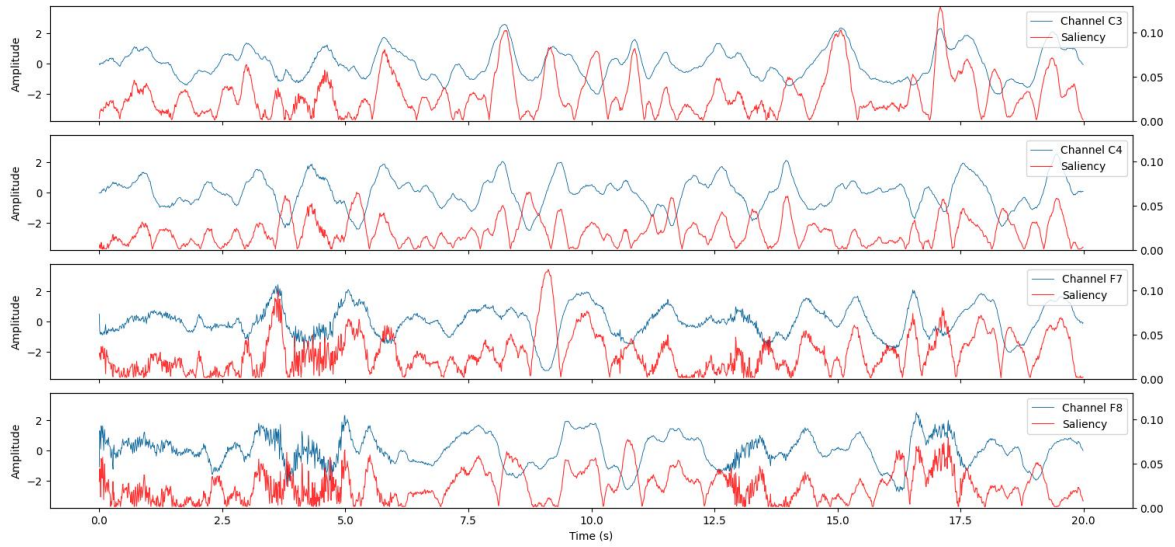


Figure C.5: Magnitude-based saliency map for a 20-second EEG segment from a patient with PCPC score 3 after 12 months.

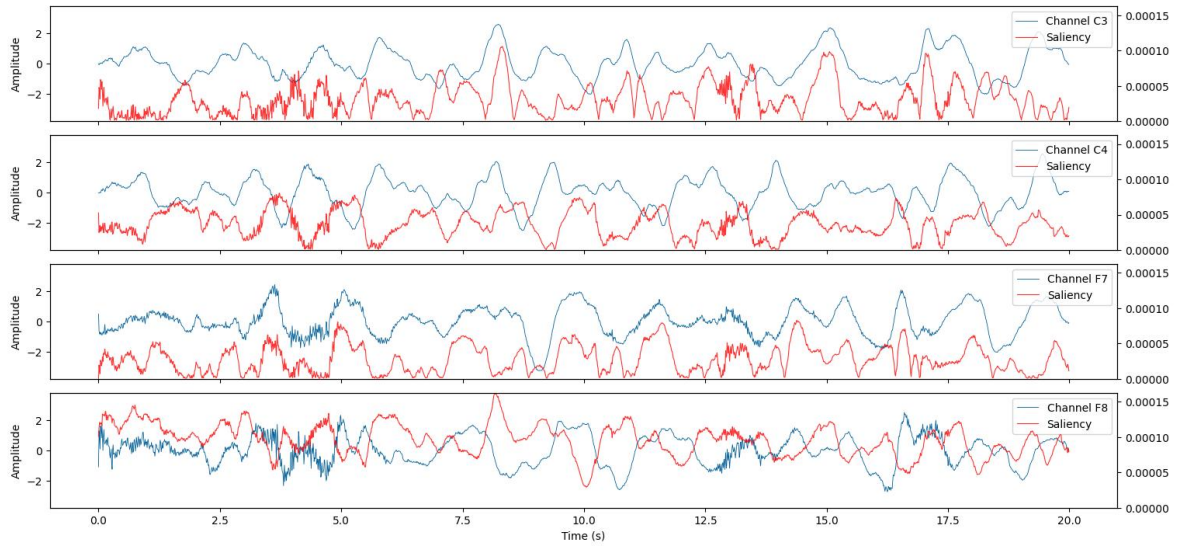


Figure C.6: Directional-based saliency map for a 20-second EEG segment from a patient with PCPC score 3 after 12 months.

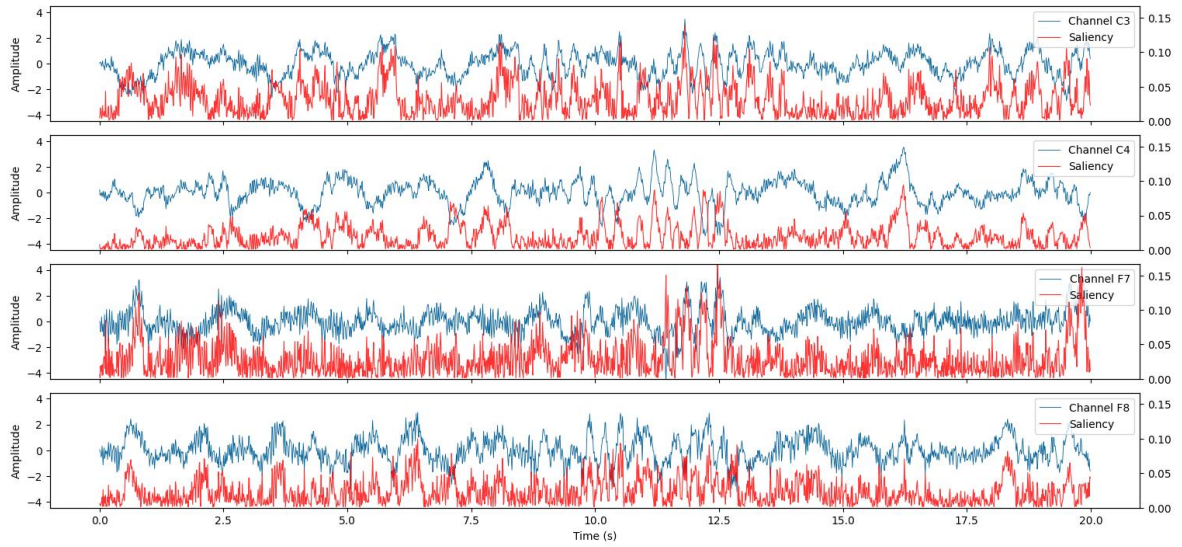


Figure C.7: Magnitude-based saliency map for a 20-second EEG segment from a patient with PCPC score 4 after 12 months.

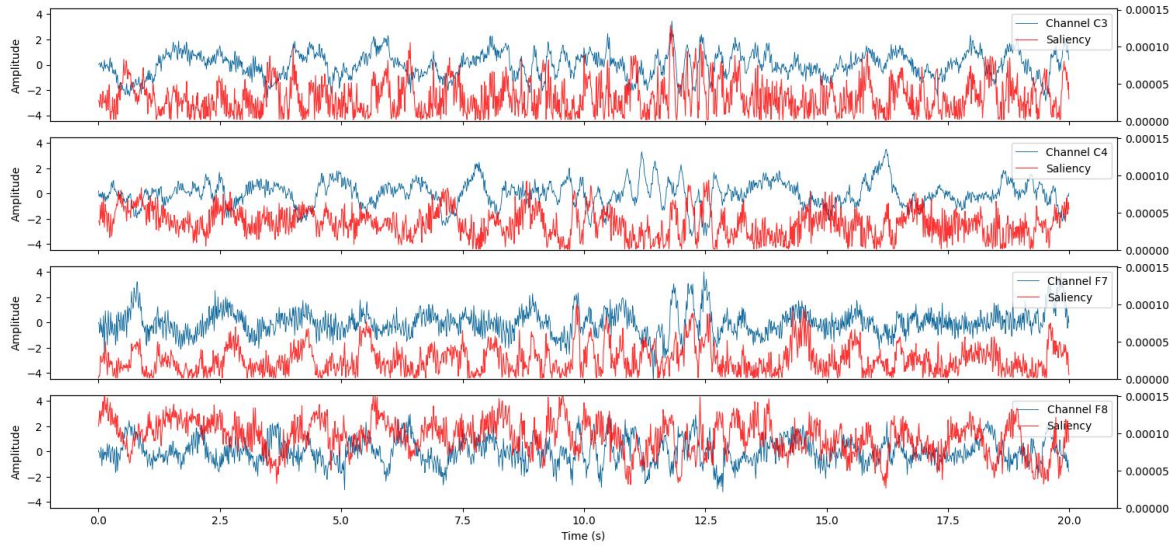


Figure C.8: Directional-based saliency map for a 20-second EEG segment from a patient with PCPC score 4 after 12 months.

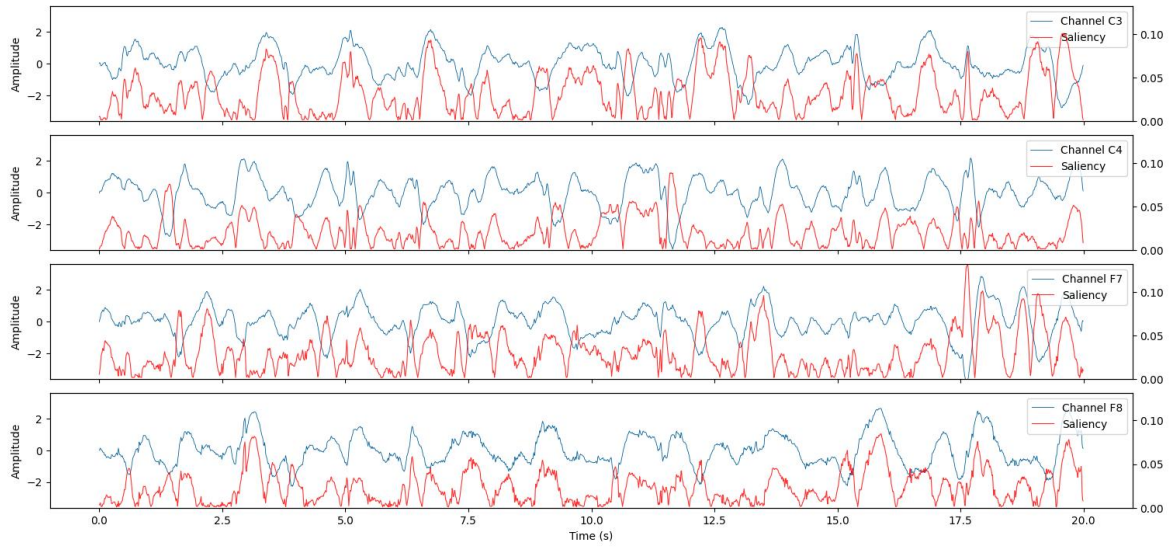


Figure C.9: Magnitude-based saliency map for a 20-second EEG segment from a patient with PCPC score 6 after 12 months.

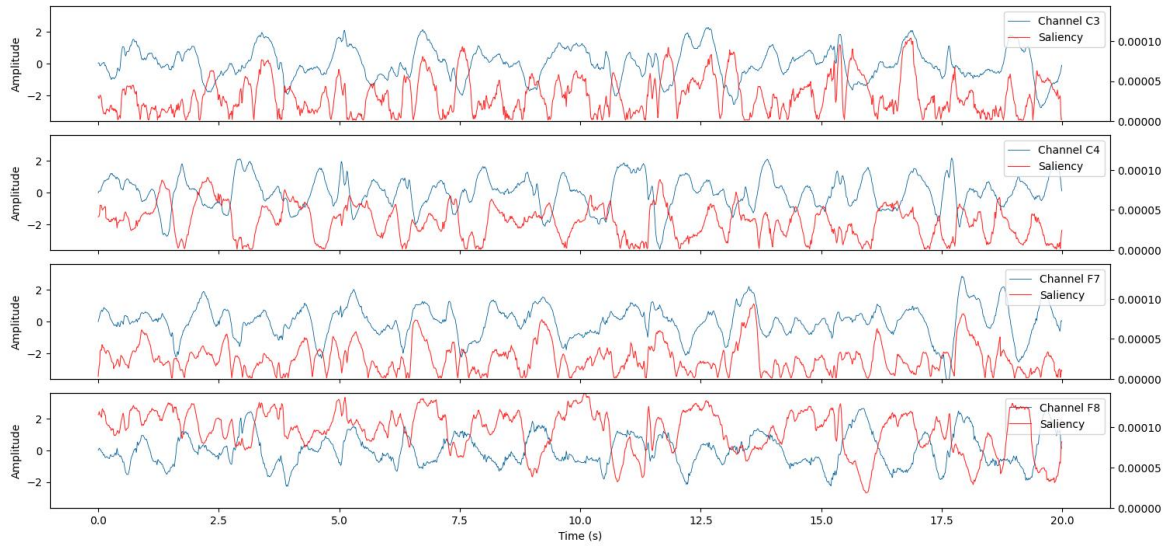


Figure C.10: Directional-based saliency map for a 20-second EEG segment from a patient with PCPC score 6 after 12 months.

D

Code and Reproducibility

All code used in this thesis is available at: `https://github.com/BerendKr/Contrastive_Learning_Classification_on_EEG_Data`

The repository includes:

- EEG preprocessing scripts (data extraction, channel selection, downsampling, normalization)
- TS2Vec-based encoder
- k-NN based Patient-level classification and evaluation
- Code for all plots reported in this thesis

Instructions for environment setup and running the pipeline are provided in the README.