Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für Integrierte Systeme
Integrated Systems Laboratory

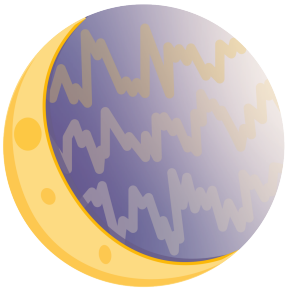
Embedded Systems Laboratory

DEPARTMENT OF
INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Fall Semester 2024

EEG Signal Analysis with Foundation Models

Master Thesis



Berkay Döner
bdoener@student.ethz.ch

07.03.2025

Advisors: Dr. Yawei Li (ETHZ), yawli@iis.ee.ethz.ch
Dr. Xiaying Wang (ETHZ), xiaywang@ethz.ch
Thorir Ingolfsson (ETHZ), thoriri@iis.ee.ethz.ch

Professors: Pr. David Atienza (EPFL), david.atienza@epfl.ch
Pr. Luca Benini (ETHZ), lbenini@iis.ee.ethz.ch

Abstract

Electroencephalography (EEG) signal modeling is important in understanding neural activity and developing diagnostic tools. The problem of variability of electrode configurations across various EEG datasets, however, presents a significant challenge. The variability of electrode configurations makes it hard to develop models that apply to any setup. This thesis overcomes the challenge by proposing a new self-supervised learning method for EEG signal modeling that efficiently achieves topology invariance in terms of EEG electrodes. This thesis aims to develop an effective model that can handle EEG data generated from varying electrode setups and easily conduct downstream tasks such as seizure detection. The primary challenge is bridging the gaps between various channel configurations and allowing models to learn essential representations regardless of particular topological features. The proposed method utilizes learned queries and cross-attention to project EEG signals from varying topologies onto a unified, fixed-size, and topology-agnostic latent space. The following temporal attention in this latent space is able to effectively capture temporal dynamics. This architecture provides a computational benefit by disentangling complexity from the number of channels. This approach not only enhances model generalizability but also introduces a computationally efficient method for learning from multi-channel EEG data, contributing to the advancement of foundation models in EEG signal processing. The method is pre-trained and fine-tuned on publicly available data and is shown to be able to perform better than or comparably to the current state-of-the-art methods in 4 downstream tasks while needing significantly lower memory and number of floating point operations.

Acknowledgments

I would like to express my gratitude to my professors, Professor Luca Benini and Professor David Atienza, for their guidance during this thesis project and for the opportunity to work on this exciting area.

I would like to especially thank my advisors, Dr. Yawei Li, Thorir Ingolfsson, and Dr. Xiaying Wang, for their continuous feedback, helpful advice, and constructive support throughout the project. Their support and encouragement greatly contributed to the completion of this thesis.

I also thank Niklas for his constant support and his valuable help in proofreading and improving the figures in this report.

Lastly, I wish to thank my colleagues Anna, Federica, and Nicolas for their support, motivation, and helpful discussions. Their positive attitude made this experience enjoyable and memorable.

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every written paper or thesis authored during the course of studies. In consultation with the supervisor, one of the following three options must be selected:

○ I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used no generative artificial intelligence technologies[1].

○ I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used and cited generative artificial intelligence technologies[2].

☑ I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used generative artificial intelligence technologies[3]. In consultation with the supervisor, I did not cite them.

Title of paper or thesis:

EEG Signal Analysis with Foundation Models

Authored by:
If the work was compiled in a group, the names of all authors are required.

Last name(s):	**First name(s):**
Döner	Berkay

With my signature I confirm the following:
- I have adhered to the rules set out in the Citation Guide.
- I have documented all methods, data and processes truthfully and fully.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for originality.

Place, date	**Signature(s)**
Zürich, 07.03.2025	~~Berkay~~

If the work was compiled in a group, the names of all authors are required. Through their signatures they vouch jointly for the entire content of the written work.

[1] E.g. ChatGPT, DALL E 2, Google Bard
[2] E.g. ChatGPT, DALL E 2, Google Bard
[3] E.g. ChatGPT, DALL E 2, Google Bard

Contents

Chapter 1

Introduction

Electroencephalography (EEG) is an important measurement technique in the field of neuroscience, offering vital information on brain activities and neurological diseases [1]. The growing availability and amount of EEG data have motivated the development of automated analysis methods, especially relying on the capabilities of deep learning [2]. Deep learning methods have revolutionized biosignal processing by enabling the ability to automatically learn complex patterns from raw data, extending beyond the limitations of traditional feature engineering methods [3]. Among these deep learning models, Transformers [4] have gained significant interest in modeling various data formats, including text [4], image [5] and audio [6]. Recent work has also shown the capabilities of transformers on multivariate time series data [7]. Their inherent ability to capture long-range dependencies and intra-variate (temporal relation within a channel) as well as inter-variate (inter-channel relations) correlations makes them highly suitable for difficult multivariate time-series signals [8] including biosignals like EEG, which exhibit both spatial and temporal structure.

However, one significant barrier still exists in EEG analysis: the inherent variability between configurations in different datasets [9]. EEG signals are registered with various electrode settings, differing in the number and position of electrodes on the scalp. This topological heterogeneity poses a significant obstacle to developing stable and clinically viable deep learning models. Topology-agnostic methods are crucial as they enable training models on heterogeneous, publicly available EEG data, leading to more generalizable and reliable models. Furthermore, some state-of-the-art approaches tend to train independent models per individual electrode setup, preventing the efficient utilization of heterogeneous datasets and the widespread use of developed models. Therefore, creating models invariant to such differences by design, capable of learning informative representations independent of the specific measurement setup, is necessary to harness the potential of deep learning for EEG. In addition to this challenge, however, the need for computational efficiency stands, especially in the clinical setup where real-time EEG analysis and diagnosis are very important. Computationally efficient models are not only necessary for real-time interventions but also to facilitate deployment on

resource-limited edge devices.

This thesis presents a new transformer modeling methodology to model topology-invariant EEG signals with a strong emphasis on computational efficiency. The proposed approach utilizes learned queries with cross-attention mechanisms to project EEG signals from varying electrode configurations onto a shared, topology-insensitive, fixed-dimensional latent space. This query-based representation is then processed with patch-wise attention to capture temporal dynamics, taking advantage of the strengths of Transformer architectures for handling time series. A key advantage of this design is that model bottleneck complexity grows with the number of patches and not with the number of electrodes, making it highly appropriate for different electrode setups and real-time use cases.

The effectiveness of the approach is assessed by self-supervised pre-training and fine-tuning on public EEG datasets. The model, with good performance on downstream tasks, e.g., seizure detection, shows the capacity to generalize across datasets with different electrode configurations. This is an important contribution to the field because it offers a topology-invariant and computationally efficient EEG modeling solution. This report is structured to provide an in-depth analysis of the aspects of the method. Chapter 2, *Background*, establishes the fundamental understanding of EEG signal characteristics, the problem posed by topological variability, the significance of deep learning, and Transformers in multivariate time series analysis. Chapter 3, *Related Work*, analyzes current approaches in EEG analysis and identifies their shortcomings in effectively tackling topology invariance and efficiency. Chapter 4, *Methodology*, explains the suggested approach, including the model architecture and the self-supervised pre-training technique. Chapter 5, *Results*, outlines the empirical assessment and performance evaluation of the model. Lastly, Chapter 6, *Conclusion*, investigates and concludes the findings.

Chapter 2

Background

2.1. Background

This chapter provides the important background knowledge necessary to understand the concepts that will be discussed later in this thesis. It will cover three key areas: an introduction to biosignals and EEG signals, their characteristics, and relevance to this work; the transformer architecture, which focuses on the attention mechanism and its components; Set Transformer methods; and the concept of learned queries.

2.1.1. Biosignals and EEG Signals

Biosignals are physiological signals that can be measured and monitored from living organisms to provide information about their biological processes [10]. These signals include a wide range of measurements, including electrical signals like EEG, electrocardiography (ECG), electromyography (EMG), or chemical signals like glucose levels. Biosignal analysis plays an increasingly crucial role in many fields, including medical diagnosis and patient monitoring [11].

Among these biosignals, EEG is a neurophysiological technique used to measure and record the brain's electrical activity [1]. It is a non-invasive method that involves placing electrodes on top of the scalp to measure the voltage differentials resulting from currents within the neurons of the brain. EEG is a valuable tool in clinical, research, and commercial settings for studying brain function, diagnosing neurological disorders, and developing brain-computer interfaces.

Recording EEG Signals

EEG signals are recorded using electrodes placed on the scalp according to standardized systems like the 10-20 system or its extensions [12]. There are different types of electrodes to measure brain activity, such as wet electrodes that require a conducting gel between the skin and the electrode discs and dry electrodes that, instead, try to eliminate the

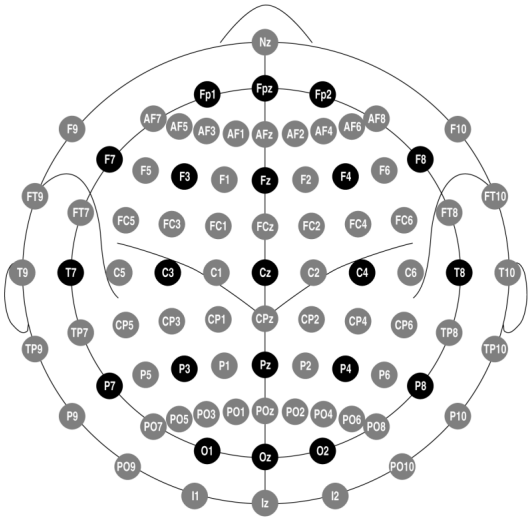


Figure 2.1.: Electrode positions and labels in the 10-20 system placed on the scalp. Gray circles indicate additional positions introduced in the 10-10 extension. [16]

need for this gel [13], each type offering some advantages and shortcomings [14]. A montage specifies the arrangement and naming convention of these electrodes. Common montages include bipolar and unipolar montages. In a referential or unipolar montage, each electrode measures the potential difference between that electrode and a common reference electrode. Meanwhile, in a bipolar montage, each channel records the potential difference between two adjacent electrodes. Electrodes are connected in chains across the scalp, and each channel represents the voltage difference between two neighboring electrodes in the chain, forming the "double-banana" montage [15].

The electrodes measure the potential difference between different locations on the scalp. These weak electrical signals are then amplified and digitized for recording and analysis. Modern EEG systems often use a multi-channel setup, recording from numerous electrodes simultaneously to capture the brain activity of different brain regions. The output is a multivariate time series, where each channel represents the electrical potential recorded at a specific electrode location over time.

Characteristics of EEG Signals

EEG signals are characterized by the following:

- **Frequency Bands:** EEG signals are typically decomposed into different frequency bands, each associated with different types of brain activity and states of consciousness [17]. The main frequency bands are:
  - **Delta (δ):** (0.5-4 Hz) - Slowest frequencies, dominant during deep sleep.
  - **Theta (θ):** (4-8 Hz) - Associated with light sleep, meditation, and memory processing.

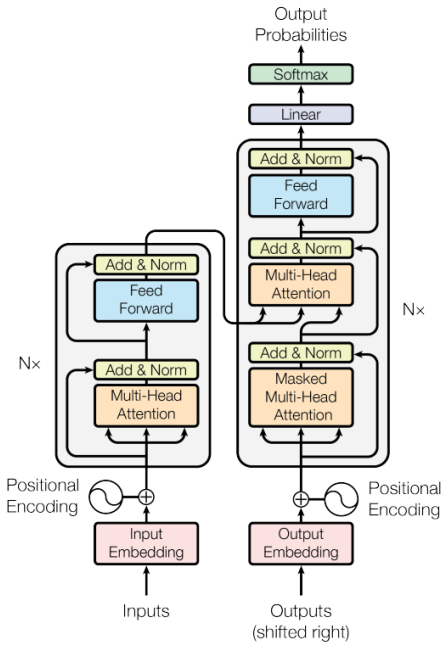


Figure 2.2.: Original transformer architecture by [4].

  - **Alpha (α):** (8-13 Hz) - Prominent when awake and relaxed with eyes closed or when sleepy.
  - **Beta (β):** (13-30 Hz) - Associated with active thinking, alertness, concentration, and motor activity and usually observed in the frontal and central regions.
  - **Gamma (γ):** (30+ Hz) - Highest frequencies associated with higher cognitive functions and sensory processing.

- **Amplitude:** The amplitude of EEG signals is typically very low, measured in microvolts (μV). The amplitude reflects the strength of the signal. Typically, EEG signals contain a lot of artifacts that require careful processing [18].
- **Multivariate Nature:** EEG is multivariate, as it simultaneously records signals from multiple electrodes. The spatial relationships between signals from different electrodes are essential for understanding activity patterns.

2.1.2. Transformer Architecture and Attention Mechanisms

The Transformer architecture, introduced by [4], has changed the natural language processing (NLP) field and has since been successfully applied to various other domains, including computer vision [5], audio analysis [6] and biosignal analysis such as ECG [19], EMG [20] and EEG [21].

At its core, the architecture relies on the **attention mechanism**, which allows the model to focus on the most relevant parts of the input sequence. This section will explain the key components of the Transformer architecture, with a particular focus on the attention mechanism.

Overall Transformer Architecture

The original Transformer architecture is primarily composed of an **encoder** and a **decoder**.

- **Encoder:** The encoder is responsible for processing the input and creating a latent representation of it. It consists of multiple identical layers. Each layer is composed of two main sub-layers:
  1. **Multi-Head Self-Attention:** This layer allows the encoder to attend to different positions in the input sequence and understand the relationships between them. For example, in the time-series domain, this would allow the model to understand the relationships between different timestamps in the input.
  2. **Feed-Forward Network:** A position-wise feed-forward network is applied to each position independently, refining the output of the attention layer.

  Each of these sub-layers is followed by a **residual connection** and **layer normalization**. Residual connections help in training deeper networks by enabling gradient flow, and layer normalization stabilizes the learning process.

- **Decoder:** The decoder is used to generate an output sequence, typically conditioned on the encoder's output. Like the encoder, the decoder is a stack of identical layers. Each decoder layer contains three sub-layers:
  1. **Multi-Head Self-Attention:** This is similar to the encoder's self-attention, but usually with masking to prevent the decoder from attending to future positions in the output sequence during training to ensure one directional generation. In the bidirectional analysis, including this thesis, masking is not applied.
  2. **Multi-Head Attention (Encoder-Decoder Attention):** This layer allows the decoder to attend to the encoder output, bridging the network's encoder and decoder parts. It takes the queries from the previous decoder layer and the keys and values from the encoder output.
  3. **Feed-Forward Network:** Identical to the encoder's feed-forward network.

  Again, each sub-layer is followed by residual connections and layer normalization. Finally, different types of layers are used at the output of the decoder to produce different task-specific outputs. For the classification task, linear layers and a softmax layer are used to predict probabilities over the classes.

The Attention Mechanism: Queries, Keys, and Values

The core of the Transformer is the **scaled dot-product attention** mechanism [4]. It relies on the calculation of an attention score between each position in the input sequence and all other positions, determining how much each position should be attended to when representing a specific position.

Defined mathematically:

- **Queries (Q):** Represent the "search term" for each position. In self-attention, queries are derived from the input sequence itself. In cross-attention, these queries come from a different source than the input sequence, which can be another sequence or the output of the encoders, etc.
- **Keys (K):** Represent the "memory term" for each position, also derived from the input sequence in self-attention.
- **Values (V):** Represent the "information content" associated with each position, again derived from the input sequence in self-attention.

For an input sequence represented as a matrix $X \in \mathbb{R}^{T \times E}$ (where T is the sequence length and E is the embedding dimension), we can obtain Query, Key, and Value matrices through linear transformations:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V$$

where $W_Q, W_K, W_V \in \mathbb{R}^{E \times d_k}$ are learnable weight matrices, and d_k is the dimension of the keys (and queries). For simplicity, we often set the query, key, and value dimensions to be the same, $d_q = d_k = d_v = d$.

The **attention score** between each query and all keys is then calculated using the dot product:

$$\text{Attention Scores} = QK^T$$

This results in a matrix of scores where each entry (i, j) represents the similarity between the i-th query and the j-th key. To stabilize training and prevent scores from becoming too large, the scores are scaled by the square root of the dimension of the keys, d_k, and **softmax function** is applied to convert them into probabilities, representing the attention weights:

$$\text{Attention Weights} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

These attention weights are then used to weigh the value vectors. The output of the attention mechanism for each query is a weighted sum of the value vectors, where the weights are the attention weights:

$$\text{Attention Output} = \text{Attention Weights} \cdot V = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

This output is a matrix of the same dimensions as Queries and Values, representing the attention-weighted representation of the input sequence.

Multi-Head Attention

To further enhance the capabilities of the attention mechanism, Transformers employ **Multi-Head Attention**. Instead of performing a single attention calculation, Multi-Head Attention runs through several independent attention heads in parallel [4].

For Multi-Head Attention, we have h different attention "heads". For each head $i \in \{1,2,...,h\}$, we learn different sets of weight matrices $W_{Q,i}, W_{K,i}, W_{V,i}$. We compute attention as described above for each head independently:

$$\text{head}_i = \text{Attention}\left(XW_{Q,i}, XW_{K,i}, XW_{V,i}\right) = \text{softmax}\left(\frac{(XW_{Q,i})(XW_{K,i})^T}{\sqrt{d_k}}\right)(XW_{V,i})$$

where $\text{head}_i \in \mathbb{R}^{T \times d_v}$. After computing the outputs for all heads, we concatenate them along the feature dimension and project them back to the original embedding dimension using another weight matrix $W_O \in \mathbb{R}^{(h \cdot d_v) \times E}$:

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1, \text{head}_2, ..., \text{head}_h)W_O$$

where Concat is the concatenation operation. Typically, the number of heads h and the dimension of each head d_v are chosen such that $h \cdot d_v = E$, the embedding dimension.

Multi-head attention helps model different aspects of the relationships between positions in the input sequence. Some heads might focus on short-range dependencies or different types of semantic or syntactic relationships.

2.1.3. Set Transformer Methods and Learned Queries

Traditional Transformer architecture is designed for sequential data, where the order of elements is crucial. The order is represented by the positional encoding that can be a mathematical function or a learned parameter based on the positional indices. However, in some scenarios, inputs are instead sets of items where the order is irrelevant. This makes Set Transformer [22] methods and the concept of learned queries become relevant. These approaches aim to achieve **permutation invariance**, meaning the output of the model should be the same regardless of the order of elements in the input set. These methods are also robust in terms of the number of elements in the input set.

Permutation Invariance and Sets

A set is an unordered collection of unique elements. In machine learning, models have to be permutation invariant when dealing with set inputs. This means that if we change the order of elements in the input set, the output of the model should not change. Mathematically, for a function f operating on a set $S = \{x_1, x_2, ..., x_N\}$, and any permutation π of the indices $\{1, 2, ..., N\}$, we want:

$$f(\{x_1, x_2, ..., x_N\}) = f(\{x_{\pi(1)}, x_{\pi(2)}, ..., x_{\pi(N)}\})$$

This property is crucial when the input is inherently unordered, such as a collection of objects, points in a point cloud, or, as in EEG input, potentially the set of EEG channels in a topology-agnostic manner.

Learned Queries for Set Processing

Learned queries provide a mechanism to process sets in a permutation-invariant way using attention mechanisms. Instead of attending between elements of the input set directly in a self-attention manner, we introduce a set of **learned query vectors**. These queries are not derived from the input set itself but are learnable parameters of the model. They act as abstract "representatives" or "aggregators" of the set.

Let $X = \{x_1, x_2, ..., x_N\}$ be the input set, where each $x_i \in \mathbb{R}^E$ is an embedded representation of an element in the set. Let $Q = \{q_1, q_2, ..., q_M\}$ be a set of M learned query vectors, where each $q_j \in \mathbb{R}^E$. These queries are initialized randomly and learned during training.

To process the set X using learned queries Q, we can apply a **cross-attention mechanism**. In this context, the learned queries Q act as the **queries** in the attention mechanism, while the elements of the input set X act as both **keys** and **values**.

For each learned query $q_j \in Q$, we compute attention over all elements in the input set X. Let's denote the matrices formed by the queries and input set elements as $Q_{mat} \in \mathbb{R}^{M \times E}$ (rows are q_j) and $X_{mat} \in \mathbb{R}^{N \times E}$ (rows are x_i). We can then perform cross-attention as follows:

$$\text{Output}_j = \sum_{i=1}^{N} \alpha_{ji} x_i \tag{2.1}$$

where α_{ji} is the attention weight between the j-th query q_j and the i-th input element x_i, calculated using scaled dot-product attention:

$$\alpha_{ji} = \frac{\exp(\text{similarity}(q_j, x_i))}{\sum_{k=1}^{N} \exp(\text{similarity}(q_j, x_k))} \tag{2.2}$$

and a common similarity function is the scaled dot product:

$$\text{similarity}(q_j, x_i) = \frac{q_j^T W'_Q (x_i W'_K)^T}{\sqrt{d_k}}$$

with learnable weight matrices $W'_Q, W'_K \in \mathbb{R}^{E \times d_k}$. For simplicity, we can directly use q_j as query and x_i as key/value and the similarity becomes:

$$\text{similarity}(q_j, x_i) = \frac{q_j^T x_i}{\sqrt{d_k}}$$

The output for the j-th query, Output_j, is a weighted sum of the input set elements, where the weights are determined by the attention mechanism with respect to q_j. Repeating this for all queries in Q gives us a set of output representations, $\{\text{Output}_1, \text{Output}_2, ..., \text{Output}_M\}$, each representing a different "view" or aggregation of the input set, guided by the learned queries.

Permutation Invariance in Learned Query Approach

The permutation invariance arises because the attention mechanism aggregates information from the entire input set for each query. Since the summation in equation (2.1) and the softmax normalization in (2.2) are permutation invariant operations with respect to the set $X = \{x_1, x_2, ..., x_N\}$, the order of elements in X does not affect the final output set of query representations $\{\text{Output}_1, \text{Output}_2, ..., \text{Output}_M\}$. Regardless of the order in which the elements of X are presented, each learned query will attend to the same set of elements and produce a consistent output representation.

In the context of the EEG signals, using learned queries allows us to create a representation of the EEG channels that is independent of their specific ordering or even the number of channels. The learned queries act as a fixed set of basis vectors to summarize the information in the input EEG electrodes, achieving topology invariance.

2.1.4. Self-Supervised Learning

Supervised learning has been widely used and a successful approach in various domains of machine learning, but it heavily relies on large amounts of labeled data. Obtaining labeled data can be expensive, time-consuming, and sometimes even infeasible, especially in domains like medical signal processing requiring expert annotations. Self-supervised learning (SSL) offers a powerful alternative by enabling models to learn meaningful representations from large amounts of unlabeled data. The core idea of SSL is to define a pretext task that can be solved using the unlabeled data itself, and learning to solve this pretext task forces the model to learn useful features that can then be transferred to downstream tasks of interest.

Leveraging Unlabeled Data with Self-Supervision

Self-supervised learning exploits the inherent structure within the unlabeled data to create its own supervisory signals. Instead of relying on external labels, SSL methods automatically generate labels from the data itself. By training a model to predict these self-generated labels, the model learns to understand the underlying patterns in the data. This pre-training phase allows the model to initialize its weights with meaningful

representations, which can significantly improve performance and reduce the need for labeled data when fine-tuning on downstream tasks.

Self-Supervised Learning in Time Series Domain

In the time-series domain, self-supervised learning has gained significant interest with the use of data-hungry transformer models, offering solutions to the challenge of limited labeled time-series data. Two main categories of self-supervision techniques for time series are masked reconstruction and contrastive learning.

- **Masked Reconstruction:** Inspired by masked language modeling in NLP [23] and masked image modeling [24] in computer vision, masked reconstruction in time series involves masking or removing portions of the input time series and training a model to reconstruct the missing parts. This pretext task forces the model to learn temporal dependencies and understand the context within the time series to accurately fill in the masked segments [25]. For example, in EEG signals, one might mask out segments of the time series in some channels and train the model to predict the original signal in the masked regions based on the context from unmasked segments and other channels.
- **Contrastive Learning:** Contrastive learning methods aim to learn representations by contrasting similar and dissimilar pairs of data samples. In the time series domain, this can be achieved by defining "positive" pairs as different views or augmentations of the same original time series segment and "negative" pairs as views from different time series segments. The model is trained to bring the representations of positive pairs closer together in the embedding space while pushing the representations of negative pairs further apart [26].

Both masked reconstruction and contrastive learning have proven effective for pre-training time series models, enabling them to learn useful representations from unlabeled data that can be effectively transferred to various downstream time series tasks. Out of these options, masked reconstruction is used in this work to pre-train models.

2.1.5. Datasets

A large EEG corpus was used to pre-train the model, and other datasets were used to measure its performance on different downstream tasks to facilitate self-supervised learning.

The primary dataset used in this project is the Temple University Hospital (TUH) EEG Corpus (TUEG), which, as described by [27], is the largest publicly available corpus of clinical EEG data. Continually collected since 2002, it comprises over 60,000 EEG recordings, amounting to more than 30 years of recording data when aggregated across all channels. This extensive corpus includes data from over 10,000 unique subjects, with an approximately balanced gender distribution and a wide age range from 1 to 90 years old. Importantly, all recordings within the TUEG dataset follow the International

10-20 system for electrode placement, ensuring a standardized channel montage. In the context of this project, the TUEG dataset serves as the primary corpus for pre-training the model, leveraging its vast amount of unlabeled EEG data.

Another dataset used for pre-training is the publicly available Siena Scalp Database from PhysioNet [28] introduced by [29], collected at the University of Siena, Italy. The primary objective of this database is to facilitate research on the description, detection, and short-term prediction of epileptic seizures and wake-sleep transitions using non-invasive EEG recordings [30]. The dataset comprises scalp EEG recordings from 14 epileptic patients (9 males, 5 females) with ages ranging from 20 to 71 years. EEG data was acquired using a video scalp EEG system with a sampling rate of 512 Hz, and electrodes were placed according to the international 10-20 system.

Beyond the comprehensive TUEG dataset, the corpus also provides labeled subsets that are valuable for evaluating model performance on specific downstream tasks. These key labeled subsets include:

- **Temple University Hospital Abnormal EEG (TUAB) Corpus:** TUAB is a subset of TUEG specifically annotated for abnormality detection. It contains recordings labeled as either 'normal' or 'abnormal', making it a benchmark dataset for clinical diagnostic tasks. TUAB comprises 2,329 subjects and features relatively balanced classes, making it suitable for training and evaluating binary classification models focused on identifying abnormal EEG patterns.

- **Temple University Hospital Artifact EEG (TUAR) Corpus:** TUAR is designed for artifact detection research. It includes annotations for various types of artifacts commonly found in EEG recordings, such as eye blinks and muscle artifacts. The annotations are provided in both single-channel and multi-channel settings. The TUAR dataset includes data from 213 subjects and is valuable for developing and testing algorithms to identify and remove artifacts from EEG signals automatically.

- **Temple University Hospital Slowing (TUSL) Corpus:** TUSL focuses on detecting and classifying EEG slowing events. It is a 4-class classification dataset, where recordings are annotated into categories including 'slowing events', 'seizures', 'complex background', and 'normal EEG'. TUSL includes 38 subjects. TUSL is helpful for tasks requiring the classification of different EEG patterns related to slowing and other neurological events.

To evaluate the broader applicability of the method beyond seizure-related EEG analysis and to assess its performance on datasets with different electrode topologies, SEED-V [31] is also used from the SEED family of datasets. These datasets are widely used benchmarks in the field of EEG-based emotion recognition research.

The SEED datasets are designed for emotion recognition from EEG signals in response to movie clips. Dataset details are as follows:

- Participants are presented with carefully selected movie clips designed to evoke specific emotions. These clips are typically categorized into discrete emotion labels.

- EEG signals are recorded from participants while they watch these movie clips. The electrode montage and number of channels vary slightly across the datasets, but all are multi-channel EEG recordings.
- Each dataset provides emotion labels for each movie clip segment. The emotion categories and the labeling methodology may differ slightly between versions (described below).

The primary task associated with the SEED datasets is to classify a participant's emotional state based on their EEG signals recorded during movie clip viewing. This is framed as a multi-class classification problem, where the classes correspond to the different emotion categories associated with the movie clips. SEED family differs by the number of emotions they categorize; the SEED dataset [32, 33], categorizes 3, the SEED-IV [34] dataset categorizes 4, and the SEED-V [31] categorizes 5 emotions, with increasing complexity and fine-grained classification. In this work, the SEED-V dataset is used to compare with the previous work.

The following table summarizes the key characteristics of the datasets used. The total number of samples is reported for the pre-training dataset, and the train/validation/test split samples are reported for the downstream datasets.

Dataset	# Subjects	# Samples	Hours of Recordings	# Channels
TUEG	14,987	15,686,874	21,787.32	20 or 22 in bipolar format
Siena	14	101,520	141.0	29
TUAB	2,329	591,357/154,938/74,010	1,139.31	22 in bipolar format
TUAR	213	49,241/5,870/5,179	83.74	22 in bipolar format
TUSL	38	16,088/1,203/2,540	27.54	22 in bipolar format
SEED-V	15	43,328/43,360/31,056	32.70	62

Table 2.1.: Summary of Datasets Used.

Chapter 3

Related Work

3.1. EEG Foundation Models

BENDR [35] is a self-supervised EEG foundation model inspired by wav2vec 2.0. BENDR uses a deep convolutional encoder to downsample raw EEG waveforms into a sequence of tokens (BENDR tokens). A Transformer encoder then processes these tokens. Positional information is added via grouped convolutions, making the model sequence length independent. BENDR is pre-trained with a Masked Autoencoding (MAE) objective: randomly masking and reconstructing BENDR tokens using a contrastive loss. BENDR is an an early promising EEG foundation model, achieving over 90% accuracy on brain-computer interface focused datasets. However, it has limitations in explicitly modeling spatial correlations between EEG channels, and the method lacks channel-specific embeddings.

BrainBERT [36] introduces a self-supervised Transformer model for intracranial EEG (iEEG), adapting techniques from speech processing. It processes iEEG signals by first converting each electrode's signal into spectrograms, which are then fed into a Transformer encoder after masked spectrogram reconstruction pre-training. This approach achieves strong performance in decoding tasks and demonstrates generalization across subjects and electrode locations on the dataset that the authors collected. BrainBERT processes each channel's signal independently to create spectrogram inputs, aiming for electrode-agnostic representations but not explicitly addressing variable electrode topologies. In contrast to this channel-independent spectrogram processing, the proposed model in this work directly tackles the challenge of varying EEG topologies.

Brant [37] proposes a large-scale foundation model for intracranial EEG pre-training, aiming to capture both long-term temporal dependencies and spatial correlations. Brant processes input by converting raw signals into patches and incorporating both time and frequency domain information through a novel encoding. The architecture comprises two Transformer encoders: a temporal encoder processing patches sequentially within each channel and a spatial encoder that subsequently captures correlations across channels. Pre-training is performed via a masked autoencoding approach, reconstructing

masked patches from the input. Evaluated on a diverse set of downstream tasks on the privately collected dataset, Brant achieves state-of-the-art performance, such as a seizure detection performance with over 0.63 F1-score. While Brant effectively incorporates spatial correlation, its architecture is inherently topology-dependent, designed for a consistent number of input channels, differing from the topology-agnostic approach.

Another method, BIOT [38], introduces a cross-data learning framework for diverse biosignals like EEG. BIOT introduces a tokenization module that processes each channel independently, segmenting it into fixed-length tokens and flattening them into a unified sequence to handle variable lengths and mismatched channels. This sequence of channel tokens is combined with the channel and positional embeddings and is then processed by a computationally efficient Linear Transformer encoder. BIOT is pre-trained using an unsupervised masked autoencoding objective and demonstrates strong performance across various tasks. The authors reported a maximum performance of 0.879 AUC-PR and 0.881 AUROC score on the TUAB dataset. While BIOT effectively addresses data heterogeneity and incorporates channel information via embeddings within its tokenized sequence, it differs from the topology-agnostic approach as it does not unify different topologies.

One of the recent foundation models, LaBraM [39], presents a large-scale foundation model for EEG, pre-trained on an extensive dataset. LaBraM addresses EEG data heterogeneity by segmenting EEG signals into channel-specific patches. These patches are then processed by a neural tokenizer trained with a vector-quantized codebook to generate compact neural codes. The core architecture is a Transformer network pre-trained with a masked signal modeling objective, predicting masked neural codes. LaBraM is designed to capture both long-term temporal dependencies and spatial correlations, utilizing separate temporal and spatial encoders in its architecture. LaBraM is the current state-of-the-art method for many datasets with high performances, such as 0.92 AUC-PR and 0.916 AUROC scores on TUAB and 0.41 accuracy on the SEED-V dataset. While achieving state-of-the-art performance on some EEG tasks and demonstrating the benefits of large-scale pre-training, LaBraM, unlike the topology-agnostic model, applies attention across a long, combined sequence of patches from all channels. This approach, although capturing spatial relationships, results in significant computational costs in terms of time and memory, particularly on data with many channels.

EEGFormer [40], is an EEG foundation model pre-trained on the TUH Corpus, aiming for transferable and interpretable EEG representations. The model employs a vector-quantized neural spectrum prediction pre-training strategy like LaBraM. EEG signals are segmented into channel patches, and a neural tokenizer is trained to encode these patches into discrete neural codes by predicting their Fourier spectrum. A Transformer encoder-decoder architecture is then pre-trained to reconstruct these neural codes from masked EEG patches. EEGFormer emphasizes interpretability through its discrete codebook and demonstrates strong performance across various downstream EEG tasks. EEGFormer achieved high scores, such as 0.852 AUROC score on the TUAR dataset. Similar to BrainBERT, EEGFormer works in a channel-independent way and does not aim to unify different topologies.

A recent work, CBraMod [41], introduces a novel foundation model for EEG decoding,

pre-trained on the TUH EEG Corpus. EEG signals are segmented into channel patches, and the architecture employs a criss-cross transformer backbone. This transformer features parallel spatial and temporal attention mechanisms to model capture spatial correlations within time intervals and temporal dependencies within channels separately, separating attention heads for both dimensions. CBraMod utilizes asymmetric conditional positional encoding (ACPE) to incorporate positional information, dynamically adapting to diverse EEG formats. Pre-training is achieved through patch-based masked EEG reconstruction. A key novelty is the criss-cross attention mechanism designed to capture EEG-specific spatial-temporal dynamics better. CBraMod, together with LaBraM, has state-of-the-art results on many datasets, such as 0.55 accuracy on the 9-class FACED emotion detection dataset and 0.915 AUROC on the TUAB dataset. The attention operation on the channel dimension might become a performance bottleneck when working with datasets with a high number of channels.

MMM [42] is a framework designed for learning topology-agnostic EEG representations. MMM addresses heterogeneity by mapping diverse channel montages into a unified scalp topology divided into predefined regions. The model utilizes region-wise tokens, representing aggregated features from channels within each scalp region, and employs a Transformer-based architecture with multi-dimensional position encoding to incorporate geometric sensor information. Pre-training is achieved through a masked autoencoding objective, reconstructing masked region-wise tokens. While MMM demonstrates state-of-the-art results in emotion recognition tasks such as over 95% accuracy on the SEED dataset and effectively achieves topology agnosticism through its unified topology mapping, it relies on hand-engineered Differential Entropy (DE) features as input rather than directly processing raw EEG signals and does not report the results on using raw signals. This makes it not an end-to-end foundation model in the same vein as models operating directly on waveforms or spectrograms. Furthermore, the paper's evaluation is primarily focused on emotion classification tasks. It also relies on predefined brain regions to form the topology.

PopT (Population Transformer) [43] proposes a modular framework for learning population-level EEG representations designed for variable electrode setups. It leverages pre-trained channel-independent temporal embeddings, which are then aggregated using a Transformer-based PopT module incorporating 3D electrode position information for spatial information. Pre-trained with discriminative self-supervised objectives (ensemble-wise and channel-wise tasks), PopT demonstrates improved decoding accuracy and generalization. While innovative in handling channel ensembles within a known topology, PopT's reliance on pre-computed temporal features distinguishes it from an end-to-end, topology-agnostic foundation model. Moreover, PopT achieves lower scores than the end-to-end models, such as a 0.882 AUROC score on TUAB, which shows the importance of end-to-end training.

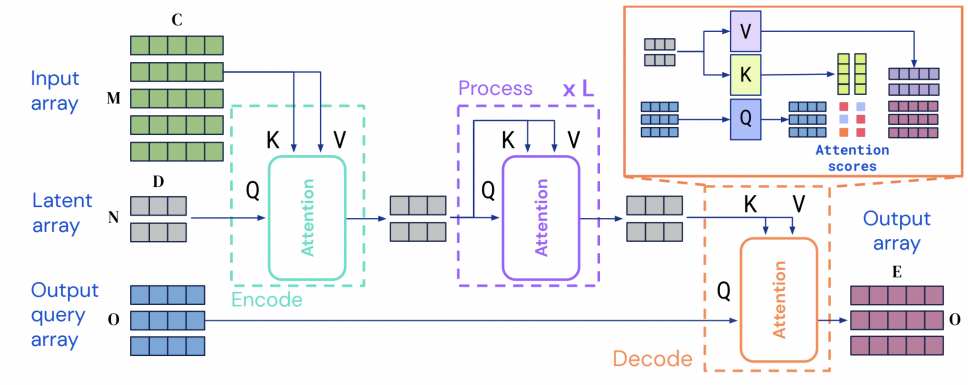


Figure 3.1.: Perceiver-IO architecture.

3.2. Set Transformer Models

Set Transformer [22], introduced in 2019, is a neural network architecture designed for permutation-invariant processing of set-structured data, addressing limitations of sequence-sensitive transformer models. The Set Transformer employs attention mechanisms within both its encoder and decoder to model interactions between set elements. They propose several blocks, which include the Set Attention Block (SAB), which applies self-attention within the set to capture pairwise and higher-order interactions, and the Induced Set Attention Block (ISAB), which reduces the quadratic complexity of self-attention to linear through the use of inducing points, enabling scalability. These inducing points are essentially learned queries for attention. The standard self-attention mechanism, as used in SAB, has a quadratic time complexity of $O(n^2)$, where n is the number of elements in the input set. The Induced Set Attention Block (ISAB) addresses this by introducing m inducing points, where m is a hyperparameter typically chosen to be significantly smaller than n ($m \ll n$). This reduces the complexity to $O(mn)$. The Set Transformer is theoretically proven to be a universal approximator for permutation-invariant functions and demonstrates empirically superior performance compared to pooling-based architectures.

Another similar method is PerceiverIO [44], which was designed as a model to work with variable-sized inputs and outputs and, thus, different modalities. The core idea lies in the flexible querying mechanism that enables the decoding of arbitrarily sized and structured outputs directly from a fixed-size latent space. Inputs of various modalities and shapes, such as images, language, audio, and symbolic sets, are encoded into this modality-agnostic latent space with cross-attention with learned queries. The architecture then leverages task-specific output queries, incorporating positional or modality embeddings, to similarly attend to this latent space. This approach decouples the computational complexity from input/output size, achieving linear scaling and enabling strong performance across a wide range of domains, including language, vision, and multimodal tasks.

Chapter 4

Implementation

4.1. Model Architecture

The proposed model, named LUNA: **L**atent **U**nified **N**etwork **A**rchitecture, is an architecture consisting of an encoder and a flexible decoder, which can achieve topology-invariant EEG signal modeling. The encoder processes the input EEG signal to generate an intermediate representation, while the decoder is task-dependent and configured for either reconstruction or classification.

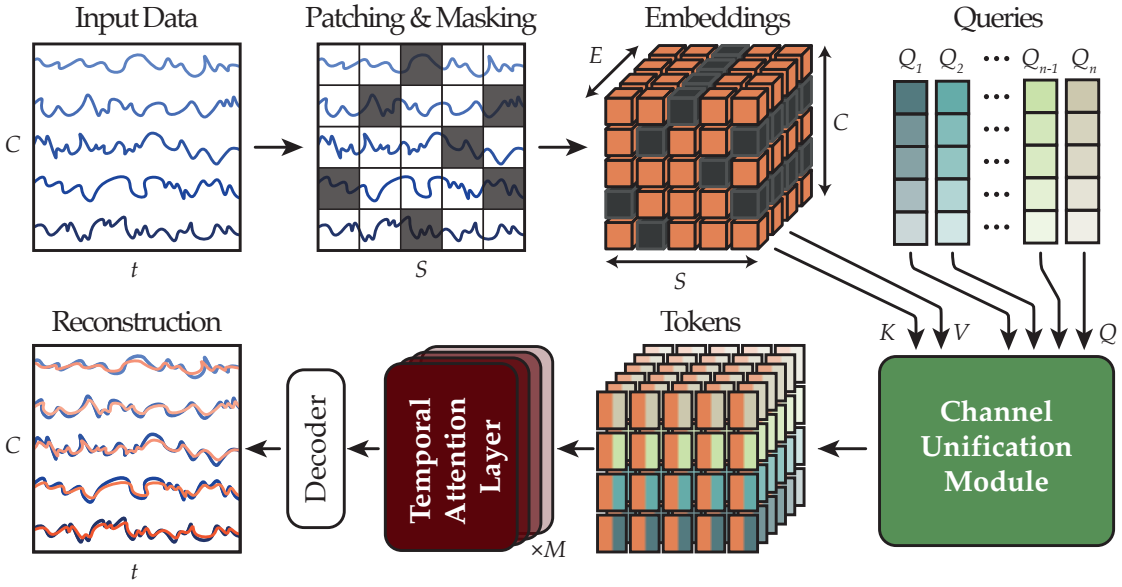


Figure 4.1.: LUNA: **L**atent **U**nified **N**etwork **A**rchitecture.

4.1.1. Encoder Architecture

The encoder transforms the input EEG signal into a unified latent space. It consists of three main stages: input patching and embedding, a cross-attention module with learned queries, and a patch-wise attention encoder.

Input Patching and Embedding The input EEG signal with shape $B \times C \times T$ is first processed to be patched and :

1. **Patching:** The input time series x of shape $B \times C \times T$ is patched along the time dimension T into non-overlapping patches (default 40 timestamps). This operation can be represented as:

$$x \in \mathbb{R}^{B \times C \times T} \xrightarrow{\text{Patching}} x_{patched} \in \mathbb{R}^{B \times (C \cdot S) \times P}$$

   where P is the patch size and $S = T//P$ is the number of patches per channel.

2. **Patch Embedding:** The patches are passed through a 3-layer network with 1D convolution layers over the patch dimension with group normalization [45] and GELU [46] activation layers in between, similar to previous work such as [39] and [41]. This module aims to map the input into a fixed embedding dimension independent of the patch size and extract meaningful features from the raw EEG signals, which are often noisy as discussed in background section Chapter 2. This module results in the embedded patch features x_{embed}:

$$x_{patched} \in \mathbb{R}^{B \times (C \cdot S) \times P} \xrightarrow{\text{Patch Embedding}} x_{embed} \in \mathbb{R}^{B \times (C \cdot S) \times E}$$

3. **Frequency Feature Embedding:** In addition to the raw signals, frequency features are also extracted for each patch. First, the real-valued Fourier Transform (FFT) is applied along the patch dimension P for each channel and patch. The magnitude and phase of the rFFT are concatenated to form frequency features. These features are then projected to the embedding dimension (E) using a multi-layer perceptron (MLP). The output of this module is the embedded patch representation:

$$x_{patched} \in \mathbb{R}^{B \times (C \cdot S) \times P} \xrightarrow{\text{Frequency Embedding}} x_{freq} \in \mathbb{R}^{B \times (C \cdot S) \times E}$$

   These features are added to the raw features:

$$x_{features} \in \mathbb{R}^{B \times (C \cdot S) \times E} = x_{embed} + x_{freq}$$

4. **Masking:** Only for the pre-training task, some patches are randomly masked, which are then predicted by the decoder for the reconstruction task. Masking is done by replacing the features of random patches with the mask embedding m_{emb}, which is a learned parameter optimized during the training. This is done by generating a binary mask $M \in \{0,1\}^{B \times (C \cdot S) \times 1}$. The masking ratio is selected to be

0.5, and the binary mask is generated using the uniform distribution. The masking ratio was selected to follow the previous work LaBraM [39] and CBraMod [41], which both showed that 0.5 gives a good balance between robustness and amount of information.

$$x_{masked_{b,n}} = \begin{cases} m_{emb} & \text{if } M_{b,n} = 0 \\ x_{features_{b,n}} & \text{if } M_{b,n} = 1 \end{cases}$$

5. **Channel Positional Encoding:** Learned channel position encodings are used to guide the cross-attention mechanism and provide information about the individual channels. Despite the potential variability in electrode topologies across datasets, a fixed channel index for each channel is used, which acts as channel-specific priors. The first C vectors in the learned embedding matrix $P_{channels} \in \mathbb{R}^{C_{max} \times E}$ are extracted for each input, which form the channel positional encoding for the input batch: $P_C \in \mathbb{R}^{C \times E}$.

After these steps, the tokens for the transformer blocks are formed by adding these features:

$$x_{token} \in \mathbb{R}^{B \times (C \cdot S) \times E} = x_{masked} + P_C$$

Channel-Unification Module with Learned Queries The tokens x_{token} are then fed into the cross-attention block to achieve topology invariance and map the channels to the fixed latent space represented by the queries. This block utilizes learned queries and a top-k cross-attention mechanism. The cross-attention is applied in the channel dimension; therefore, the tokens are reshaped from $B \times (C \cdot S) \times E$ to $(B \cdot S) \times C \times E$. This reshaping allows the cross-attention mechanism to work only on the channel dimension and independent from the patch dimension. It also decreases the complexity of the operation as the sequence length is reduced from the original $C \cdot S$ to C. This module uses (Q) learnable query vectors, $Q \in \mathbb{R}^{1 \times Q \times E}$, which are orthogonally initialized and act as channel aggregators. The cross-attention mechanism, detailed in the algorithm below, computes affinity scores between the learned queries and the channel features. The module uses a top-k sparse attention approach, where each channel attends to its most similar query. The queries are then updated by aggregating information from the attended channels.

Algorithm 1: Cross-Attention with Top-k Sparse Attention.

Input : Queries $Q \in \mathbb{R}^{(B \cdot S) \times Q \times E}$ `// Learnable Queries`
Keys $K \in \mathbb{R}^{(B \cdot S) \times C \times E}$ `// EEG Channel Features (Keys)`
Values $V \in \mathbb{R}^{(B \cdot S) \times C \times E}$ `// EEG Channel Features (Values)`
Temperature $T \in \mathbb{R}$ `// Temperature Parameter`
Output: Updated Queries $Q' \in \mathbb{R}^{(B \cdot S) \times Q \times E}$ `// Updated Queries`
Attention Weights $A \in \mathbb{R}^{(B \cdot S) \times Q \times C}$ `// Attention Weights`

$S = Q \cdot K^T$ `// Compute affinity scores`
$I_{topk} = \texttt{topKIndices}(S, k = 1, \dim = 1)$ `// Determine top-k query indices`
$Mask = \texttt{generateBinaryMask}(I_{topk})$ `// Create attention mask`
$S' = S \odot Mask$ `// Apply mask to the affinity scores`
$A_{q,c} = \texttt{Softmax}(S'_{q,c}/T, dim = -1)$ `// Compute attention weights`
$Q' = Q + A \cdot V$ `// Aggregate values with attention weights`
`return` Q', A

The top-k selection mechanism encourages each query to specialize in attending to and extracting features from the most relevant channels based on affinity scores. This results in a more interpretable latent space and learning more diverse features for each query. The experiments show that it is crucial for the queries to learn orthogonal patterns over the channels to reconstruct the channels successfully.

After the queries are updated through the cross-attention, they are further refined using the query attention module. This module consists of two transformer encoder layers, applying self-attention among the queries. This step allows the learned queries to interact with each other, which helps the model capture higher-order relationships within the set of query representations. By attending to each other, the queries can collectively refine their representations and create a more informative latent space over the channels.

Patch-wise Attention Encoder After the cross-attention module, the tensor is reshaped and processed by a series of transformer blocks.

- **Reshaping for Patch-wise Attention:** The output of the cross-attention module, with shape $(B \cdot S) \times Q \times E$, is reshaped to $B \times S \times (Q \cdot E)$ to prepare for patch-wise temporal attention.

$$(B \cdot S) \times Q \times E \xrightarrow{\text{Reshape}} B \times S \times (Q \cdot E)$$

- **Transformer Blocks:** The reshaped tensor is passed through `depth` (L) layers of transformer encoder blocks. These blocks apply self-attention over the patch (S) dimension, which helps the model to learn temporal dynamics across patches within the latent space represented by the learned queries. Rotary positional embeddings [47] are utilized within these blocks to encode positional information

along the patch dimension. This approach uses rotation matrices to rotate queries and keys for each item in the sequence to identify individual positions. Rotary positional embeddings are a good fit for patch-wise attention as they can generalize well to longer sequences than the pre-training samples and encode both absolute and relative positions for each token. Each encoder block consists of a layer normalization layer [48] at the beginning (similar to the Pre-LN approach described by [49]), an attention block, and a feed-forward network.

- **Layer Normalization:** A layer normalization layer is applied to the output of the final Transformer Block to stabilize the representations.

4.1.2. Decoder Architecture

The decoder architecture is designed to be flexible, which can be used to map the encoder output to different shapes through the use of learned queries. The decoder is kept small in size compared to the encoder to guide the encoder to learn more informative representations of the input, which can be adapted easily to different tasks or datasets by fine-tuning.

Flexible Decoder Design The decoder leverages learned queries to map the encoder's latent representation to task-specific outputs. Two decoder variants are implemented:

- **Reconstruction Head:** Used for the self-supervised masked signal reconstruction pre-training task.
  - **Learned Decoder Queries:** Learned queries $Q_{dec} \in \mathbb{R}^{C \times E}$ are used in the decoder, with the number of queries matching the number of input channels (C) for detailed reconstruction. This recovers the original channel space from the latent space learned by the channel-unification module. The cross-attention is applied between the reshaped encoder output $E_{enc} \in \mathbb{R}^{(B \cdot S) \times Q \times E}$ and the decoder queries $Q_{dec} \in \mathbb{R}^{C \times E}$ to produce $E_{dec} \in \mathbb{R}^{(B \cdot S) \times C \times E}$. Learned decoder queries are specific to each electrode to be reconstructed, i.e., there are as many queries as distinct electrodes in the pre-training data.
  - **Transformer Decoder:** A transformer decoder layer is used to perform cross-attention between the learned decoder queries and the encoder output.
  - **Linear Projection:** A final linear layer projects the decoder output back to the original patch size (P) for signal reconstruction.
- **Classification Head:** Used for downstream classification tasks.
  - **Single Learned Aggregation Query:** A single learned query $Q_{agg} \in \mathbb{R}^{1 \times Q \cdot E}$ is used to aggregate information from all patches for classification. This results in a better aggregation than the mean aggregation over the patch dimension. The cross-attention is applied between the encoder output $E_{enc} \in \mathbb{R}^{B \times S \times (Q \cdot E)}$ and the decoder query $Q_{dec} \in \mathbb{R}^{1 \times (Q \cdot E)}$ to produce $E_{dec} \in \mathbb{R}^{B \times 1 \times (Q \cdot E)}$.

– **Attention and MLP:** A multi-head attention layer performs attention between the single learned query and the encoder output. An MLP then processes the aggregated query representation to output classification logits.

4.1.3. Complexity Analysis of Attention Mechanisms

The proposed latent space approach brings some benefits to computational efficiency in addition to unifying the different topologies. In this section, the complexity of its key attention modules is analyzed and compared against the alternative approaches.

Complexity Analysis of Model Stages

Table 4.1 provides a breakdown of the computational complexity for each stage within the encoder architecture.

Stage	Input Shape	Complexity
Channel-Unification Module	$(B \cdot S) \times C \times E$	$\mathcal{O}(B \cdot S \cdot Q \cdot C \cdot E)$
Query Self-Attention	$(B \cdot S) \times Q \times E$	$\mathcal{O}(B \cdot S \cdot Q^2 \cdot E)$
Patch-wise Attention Encoder	$B \times S \times (Q \cdot E)$	$\mathcal{O}(B \cdot S^2 \cdot Q \cdot E)$

Table 4.1.: Complexity Breakdown of Model Stages

The number of learned queries Q is designed to be significantly smaller than the number of input channels C ($Q \ll C$), which ensures that the query self-attention stage while having a quadratic complexity in the number of queries (Q^2), does not become the computational bottleneck. Instead, the Patch-wise Attention Encoder, with complexity scaling quadratically with the number of patches S^2, is the primary factor when the number of patches grows large, for example, when processing long EEG recordings. In scenarios with extremely high channel counts, the Channel-Unification Module can have a higher complexity than the Patch-wise Attention stage. However, even in such cases, the Channel-Unification Module maintains a linear complexity scaling with respect to the number of channels, which is a significant improvement over architectures with quadratic channel complexity.

Complexity Comparison of Different Approaches

Table 4.2 compares the overall computational complexity of the proposed architecture with other alternatives used in the literature. The method is compared with the full attention and the alternating attention, which represent:

- **Full Attention:** Single self-attention mechanism over the entire flattened sequence of channel and time patches, i.e. inputs with shape $B \times (S \cdot C) \times E$. This approach was used by LaBraM [39].
- **Alternating Attention:** Decouple spatial and temporal attention by applying self-attention either patch-wise across all channels or channel-wise across all patches,

i.e. inputs with shape either $(B \cdot C) \times S \times E$ or $(B \cdot S) \times C \times E$. This approach was used by [41] and [50].

Method	Complexity
Latent Space Attention	$\mathcal{O}(B \cdot S^2 \cdot Q \cdot E)$
Full-Attention (Channel & Patch)	$\mathcal{O}(B \cdot S^2 \cdot C^2 \cdot E)$
Alternating Attention (over Patches)	$\mathcal{O}(B \cdot S^2 \cdot C \cdot E)$
Alternating Attention (over Channels)	$\mathcal{O}(B \cdot S \cdot C^2 \cdot E)$

Table 4.2.: Attention Complexity Comparison

Full attention, while directly modeling all pairwise relationships, has a high computational cost with a significant bottleneck complexity, scaling quadratically with both the number of patches (S) and channels (C). Alternating attention, while aiming to reduce the quadratic complexity of full attention, still has a quadratic scaling factor with either patches or channels, remaining computationally demanding, compared to the latent space attention, which achieves a lower bottleneck complexity by using learned queries to reduce the channel dimension before applying patch-wise attention.

4.2. Training Setup

4.2.1. Self-Supervised pre-training Task

This section outlines the details of the self-supervised pre-training task, including the definition of the task and the loss functions used.

Masked Patch Reconstruction

The self-supervised pre-training task is designed as a masked patch reconstruction task. The model is trained to reconstruct randomly masked patches of the input EEG signal, which helps the model to learn meaningful representations by using the context provided by the unmasked patches of the signal. This task enables the encoder to capture the temporal and spatial aspects of the EEG data without relying on label supervision. This pre-training task has been shown to be effective by many previous methods such as BrainBERT [36], LaBraM [39], and CBraMod [41].

Reconstruction Loss Function

The self-supervised pre-training objective is to minimize a reconstruction loss that measures the difference between the original EEG signal patches and the reconstructed patches. The loss function is a Smooth L1 Loss, computed both for masked and non-masked (visible) patches and then combined.

The total reconstruction loss ($\mathcal{L}_{rec}$) is formulated as:

$$\mathcal{L}_{reconstruction} = \mathcal{L}_{masked} + \alpha \cdot \mathcal{L}_{visible}$$

where:
$\mathcal{L}_{masked}$ is the Smooth L1 Loss computed over the masked patches:

$$\mathcal{L}_{masked} = \frac{1}{N_{masked}} \sum_{i \in \mathcal{M}} L_{\text{SmoothL1}}(x_{original_i}, \hat{x}_{reconstructed_i})$$

$\mathcal{L}_{visible}$ is the Smooth L1 Loss computed over the non-masked (visible) patches:

$$\mathcal{L}_{visible} = \frac{1}{N_{visible}} \sum_{i \notin \mathcal{M}} L_{\text{SmoothL1}}(x_{original_i}, \hat{x}_{reconstructed_i})$$

where $\mathcal{M}$ is the set of masked patches, and N_{masked} is the total number of masked patches, and $N_{visible}$ is the total number of visible patches in the batch. The weighting factor α weighs the contribution of the visible patch loss to the masked loss. Including the visible patch loss with a small weight can help stabilize training. $\alpha = 0.05$ was used during the experiments.

Smooth L1 Loss $L_{\text{SmoothL1}}(x, \hat{x})$ is defined as:

$$L_{\text{SmoothL1}}(x, \hat{x}) = \begin{cases} 0.5(x - \hat{x})^2 & \text{if } |x - \hat{x}| < \beta \\ \beta |x - \hat{x}| - 0.5\beta^2 & \text{otherwise} \end{cases}$$

where β is a hyperparameter set to 1.

Query Load Balancing Loss

In addition to the reconstruction loss, an auxiliary loss function is used to balance the utilization of the queries. Some experiments showed that the model was not utilizing the queries equally and that some queries were rarely used, which leads to redundancy and the reduced representation capacity by the queries. This is a common problem observed in the mixture-of-experts (MoE) literature as well, which is often fixed by adding an auxiliary loss function to balance the expert utilization [51]. Influenced by this loss function in the MoE domain, a load balancing loss is applied over the queries.

The loss ($\mathcal{L}_{load\_balancing}$) is mathematically formulated as:

$$\mathcal{L}_{load\_balancing} = \alpha \cdot Q \cdot \sum_{q=1}^{Q} \left(\texttt{channels\_per\_query}_q \cdot \texttt{query\_prob\_per\_channel}_q \right)$$

where $\texttt{channels\_per\_query}_q$ is the fraction of channels handled by query q,

$$\texttt{channels\_per\_query}_q = \frac{1}{C} \sum_{x \in \mathcal{B}} \mathbb{1}\{\text{argmax } A(x) = q\} \tag{4.1}$$

and $\texttt{query\_prob\_per\_channel}_q$ is the average affinity score between the query q and all channels,

$$\texttt{query\_prob\_per\_channel}_q = \frac{1}{C} \sum_{x \in \mathcal{B}} A(x). \tag{4.2}$$

α is a hyperparameter controlling the strength of the auxiliary loss, which is kept at $\alpha = 0.2$ during the experiments. The coefficient was selected to be lower than the reconstruction loss to keep the reconstruction loss as the primary guiding factor. This loss term is also minimized during pre-training, encouraging the model to distribute attention to all the queries.

4.2.2. Classification Tasks

The pre-trained encoder is adapted for downstream classification tasks in the supervised fine-tuning stage. This involves attaching a task-specific classification head to the encoder and fine-tuning the model on labeled EEG datasets.

Downstream Tasks

The downstream performance is evaluated on several classification tasks, such as abnormal EEG detection and emotion classification. The output layer of the decoder is adapted to the number of classes in each dataset. The model is fine-tuned on the labeled dataset, and its performance is evaluated using classification metrics.

Classification Loss Function

For supervised fine-tuning on classification tasks, the cross-entropy loss is used as the loss function to train the model to predict class labels. Given the predicted logits y_{preds_logits} and the ground truth labels y, the cross-entropy loss ($\mathcal{L}_{CE}$) is calculated as:

$$\mathcal{L}_{CE} = -\frac{1}{B} \sum_{b=1}^{B} \sum_{c=1}^{N_{classes}} y_{b,c} \log(\text{softmax}(y_{logits}))_{b,c}$$

Evaluation Metrics for Classification

Performance on downstream classification tasks is evaluated using the following metrics:

- **Accuracy:** Calculated as the percentage of correctly classified samples. Macro-averaged accuracy is used for multi-class tasks.
- **Recall:** Measures correctly identifying positive instances for each class. Macro-averaged recall is used for multi-class tasks.
- **Precision:** Measures the fraction of correctly predicted positive instances out of all instances predicted as positive for each class. Macro-averaged precision is used for multi-class tasks.

- **F1-Score:** The harmonic mean of precision and recall. Macro-averaged F1-score is used for multi-class tasks.
- **Cohen's Kappa:** Measures the agreement between predicted and actual labels.
- **AUC-ROC:** For binary classification tasks, AUC-ROC measures the model's ability to identify positive and negative classes across various threshold settings. Macro-averaged AUC-ROC is used for multi-class tasks.
- **Average Precision (AUC-PR):** For binary classification tasks, AUC-PR measures the weighted average of precision at each threshold, which explains the precision-recall curve. Macro-averaged AUC-PR is used for multi-class tasks.

4.3. Dataset Preprocessing and Organization

The raw EEG signals are preprocessed through some steps to ensure high data quality and prepare the data for the training.

4.3.1. Data Preprocessing

The datasets are preprocessed as follows:

1. EEG measurements from different sessions are loaded, and the subjects that are also present in the downstream datasets (TUAB, TUAR, TUSL) are excluded from the pre-training dataset to prevent data leakage.
2. Irrelevant channels, such as ECG leads, are dropped from the raw data, and the remaining channels are reordered to follow a consistent and standard order.
3. A bandpass filter is applied to the raw EEG data, filtering the signal within the 0.1Hz to 75.0Hz band. This helps the data to focus only on the relevant frequency range for EEG analysis.
4. A notch filter at 50Hz or 60Hz is applied to remove power line interference, based on where the dataset was collected.
5. The signal is resampled to 256Hz to ensure consistency across datasets. The data is resampled to 256Hz as the majority of the data is measured at this frequency.
6. Optionally, data is mapped to bipolar montage format instead of a unipolar referential approach. This is done on the TUEG dataset as well as the TUAB, TUAR, and TUSL datasets but not on Siena or SEED.
7. The preprocessed EEG data is sliced into non-overlapping segments of 5 seconds to create individual EEG samples for training. This is done to separate the very long sessions into manageable chunks, as done by previous methods such as LaBraM [39]. This is different for the SEED datasets, which have either 1-second or 4-second samples by default.

8. The preprocessed data is saved into a file. Samples with different numbers of channels are saved to different files, and the pre-training dataset is saved into 37 smaller files (4 with 20 channels, 32 with 22 channels, and 1 file with 29 channels). Reading from smaller subsets is faster than reading from a single large file.

4.3.2. Data Module for Varying Shapes

A custom data loader is designed to handle EEG datasets with varying channel configurations and optimize data loading, which is especially important for pre-training. The data module processes dataset files sequentially, preventing the problem of trying to form batches of samples with different sizes, as the individual dataset files have the same number of channels. However, the data loader also concatenates the EEG samples from different files but with the same number of channels to potentially form batches with samples coming from different files. This allows the different data loader workers to read from different files while forming batches, which makes data loading much faster than concurrent reads from the same dataset file.

Chapter 5

Results

In this section, the proposed method is compared against the alternatives in the literature in terms of both the downstream dataset performance and the speed and memory usage. Furthermore, the proposed method is analyzed based on the latent space represented by the queries, how the model handles different topologies, how the model performs on the reconstruction task, and the representations learned by the model. This work presents three models with different sizes (base-large-huge) and compares them to state-of-the-art methods.

5.1. Training Curves

The training curves for the small model can be seen in 5.1. The figures show the reconstruction loss and the query load balancing loss over the epochs. The reconstruction loss curve shows a significant drop in the loss early in the training, which is probably due to the model learning the easy patterns in the data, such as the signal range. Afterwards, the model continues to improve slowly for many epochs; however, the loss shows little to no improvement after 100 epochs. Query load balancing loss shows a similar pattern of a sharp decrease in the beginning, followed by minor improvements. The entropy loss shows some spikes during training but stabilizes quickly after 50 epochs to its minimum value of 0.2 (the coefficient for the load balancing loss is 0.2, and the minimum loss value is 1, thus resulting in a minimum overall value of 0.2).

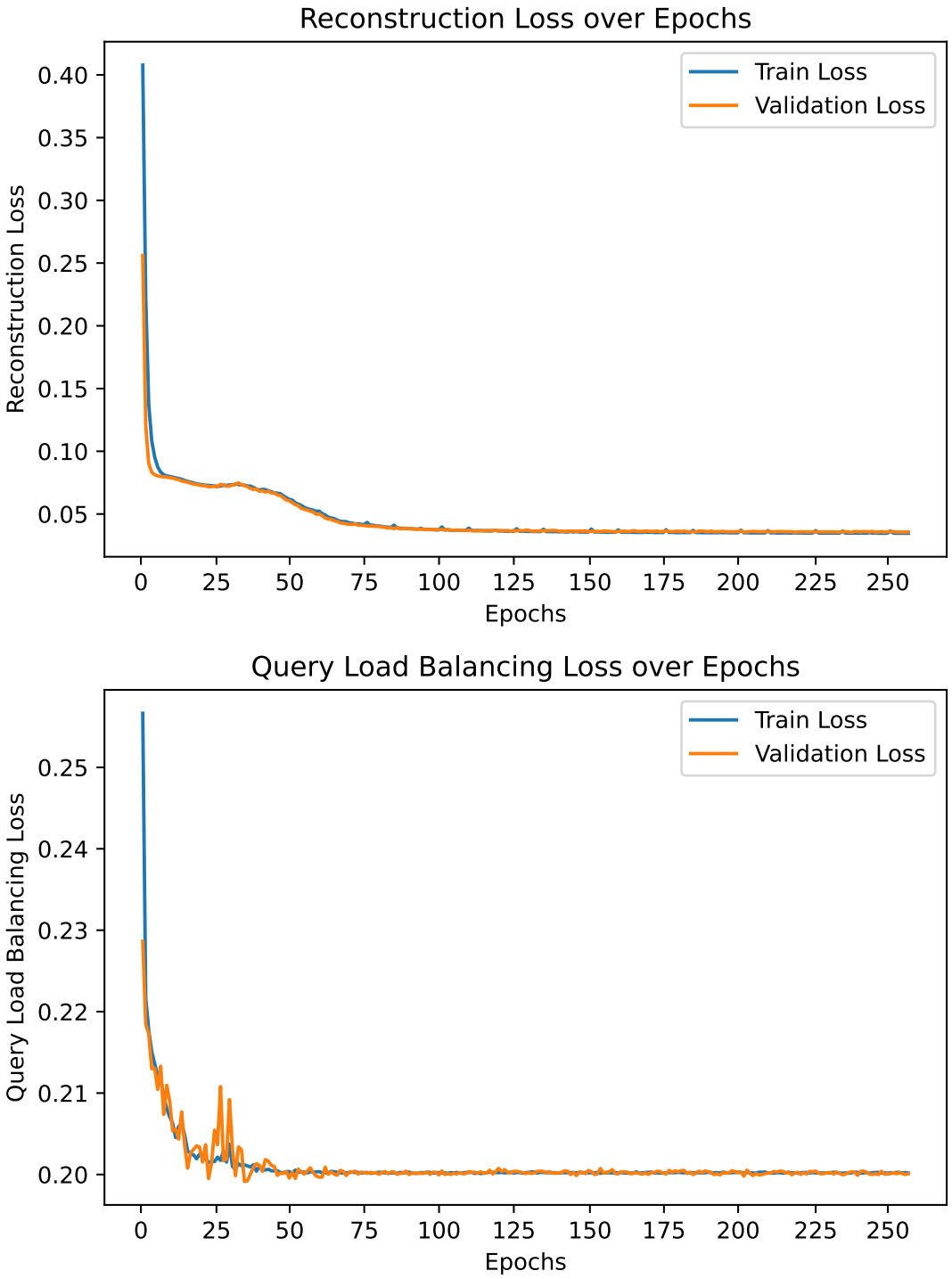


Figure 5.1.: Loss curves during pre-training.

5.2. Reconstruction Results

The reconstructions of the masked patches for samples with different topologies can be seen in Figure 5.2, Figure 5.3, and Figure 5.4. The masked regions are indicated with a gray background. As can be seen from the figures, the model can reconstruct the signals from different datasets, 20 and 22 channels data coming from the TUEG dataset and the 29 channels data coming from the Siena dataset. This confirms that the model can unify different topologies during training. Furthermore, the reconstructions of the model are much smoother than the original EEG signals, which have high noise-to-signal ratios. This shows that the model can learn representations using the underlying trend of the raw signals, not focusing on the noises.

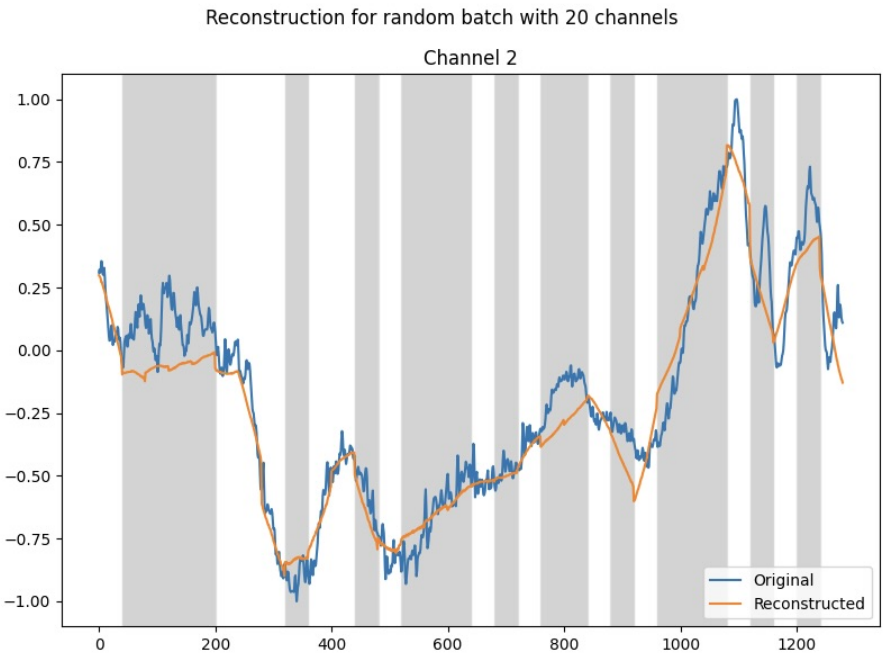


Figure 5.2.: Reconstructions on input with 20 channels.

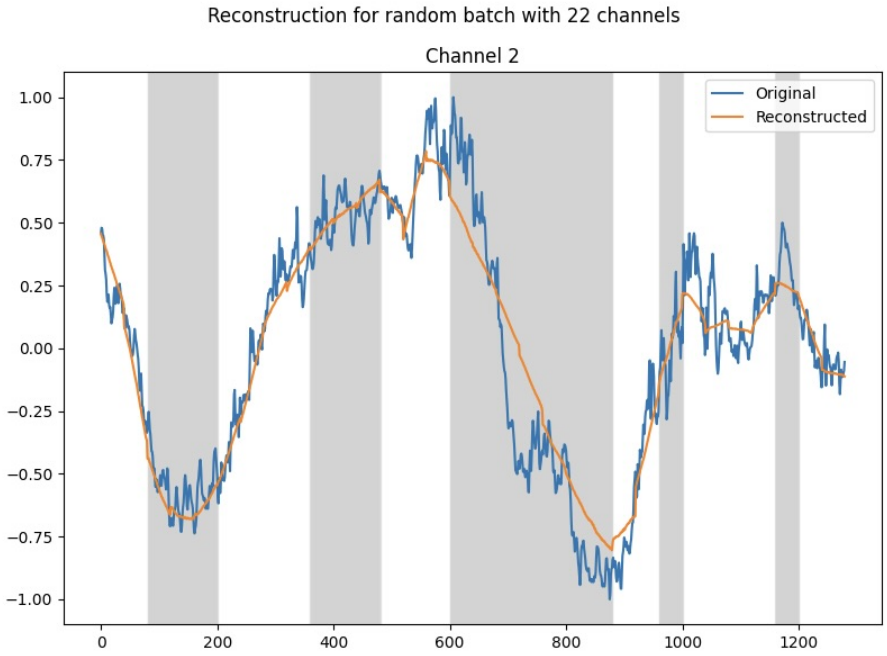


Figure 5.3.: Reconstructions on input with 22 channels.

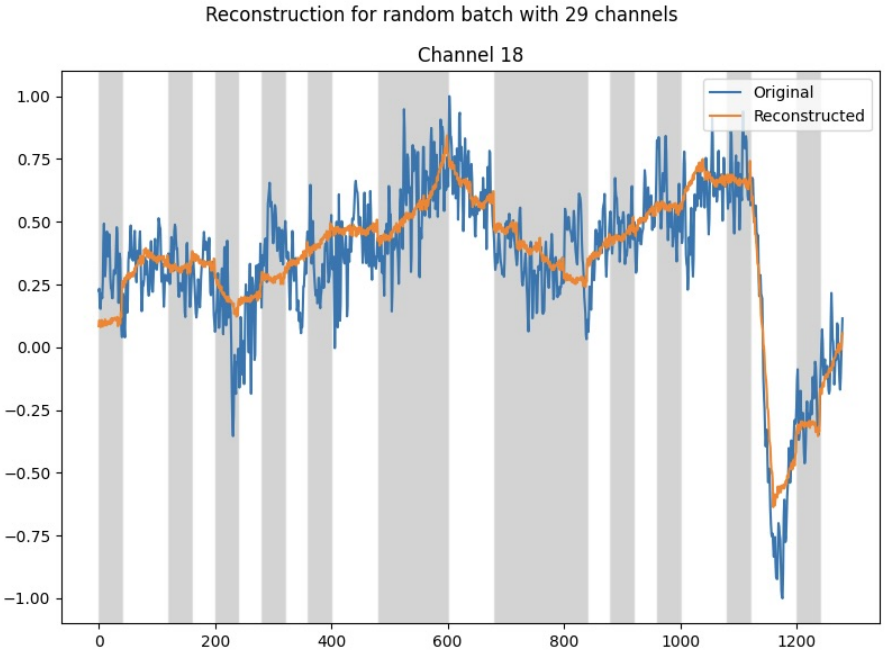


Figure 5.4.: Reconstructions on input with 29 channels.

5.3. Representation Analysis

This section analyzes the representations learned by the model on the downstream tasks. This analysis can be conducted by encoding samples from downstream datasets into the latent representation using only the encoder part of the network. These embeddings can be visualized in 2D space using dimensionality reduction and with the corresponding class labels. This shows the strength of the features learned by the model even without fine-tuning them for the downstream tasks. Figure 5.5 shows the 2D visualization of the TUAB dataset using the small model, which somewhat separates normal and abnormal signals. Figure 5.6 shows the visualization using the TUAR dataset, which shows some artifacts that affect the EEG signal. It is clear from the plot that the model can separate the electrode artifacts from the other labels. The samples with no artifacts are also somewhat clustered together. The model can be fine-tuned to refine these representations and fully adapt to different downstream datasets.

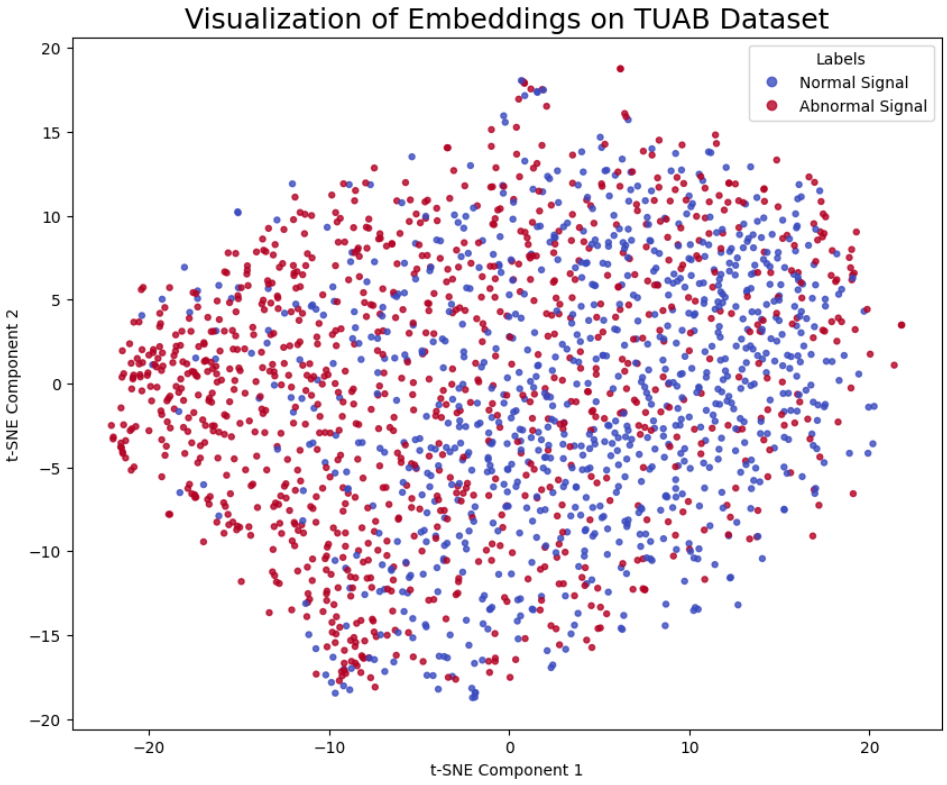


Figure 5.5.: Embeddings on the TUAB dataset.

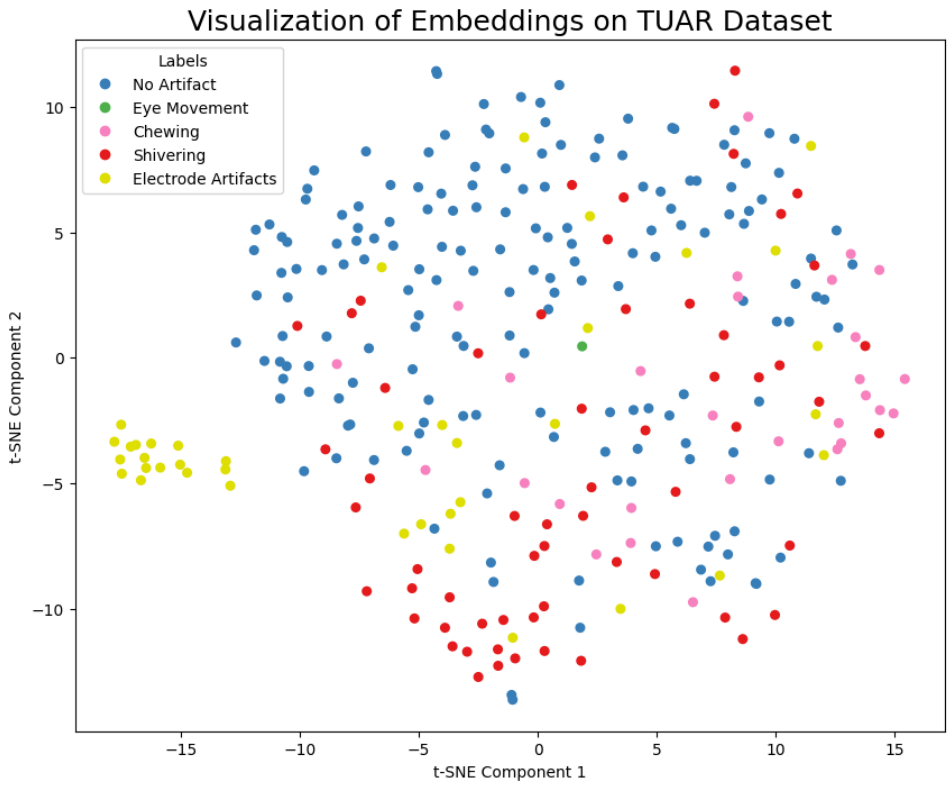


Figure 5.6.: Embeddings on the TUAR dataset.

5.4. Latent Space Analysis

Visualization of the learned queries, as depicted in Figure 5.7, reveals how the channel-unification module constructs a topology-agnostic latent space representation of the input EEG channels. Each topographic map corresponding to a specific query visualizes the average attention scores of that query across all input channels, which are averaged over some samples. The distinct patterns shown by different queries indicate that they learn to specialize in attending to and aggregating information from different but potentially overlapping subsets of EEG channels. Some queries, such as Query 3 and Query 7, show more localized attention patterns, focusing on specific brain regions like the right temporal/occipital and right frontal areas, respectively. This suggests that these queries are learning to capture localized spatial features associated with particular brain regions. Other queries, like Query 1 and Query 4, exhibit more distributed attention patterns, indicating that they may be learning to represent more global EEG features that are not confined to specific brain regions. Since there is no constraint on how queries can attend to channels in different regions, the model can learn to represent them using the same query, showing flexibility over regional aggregation approaches.

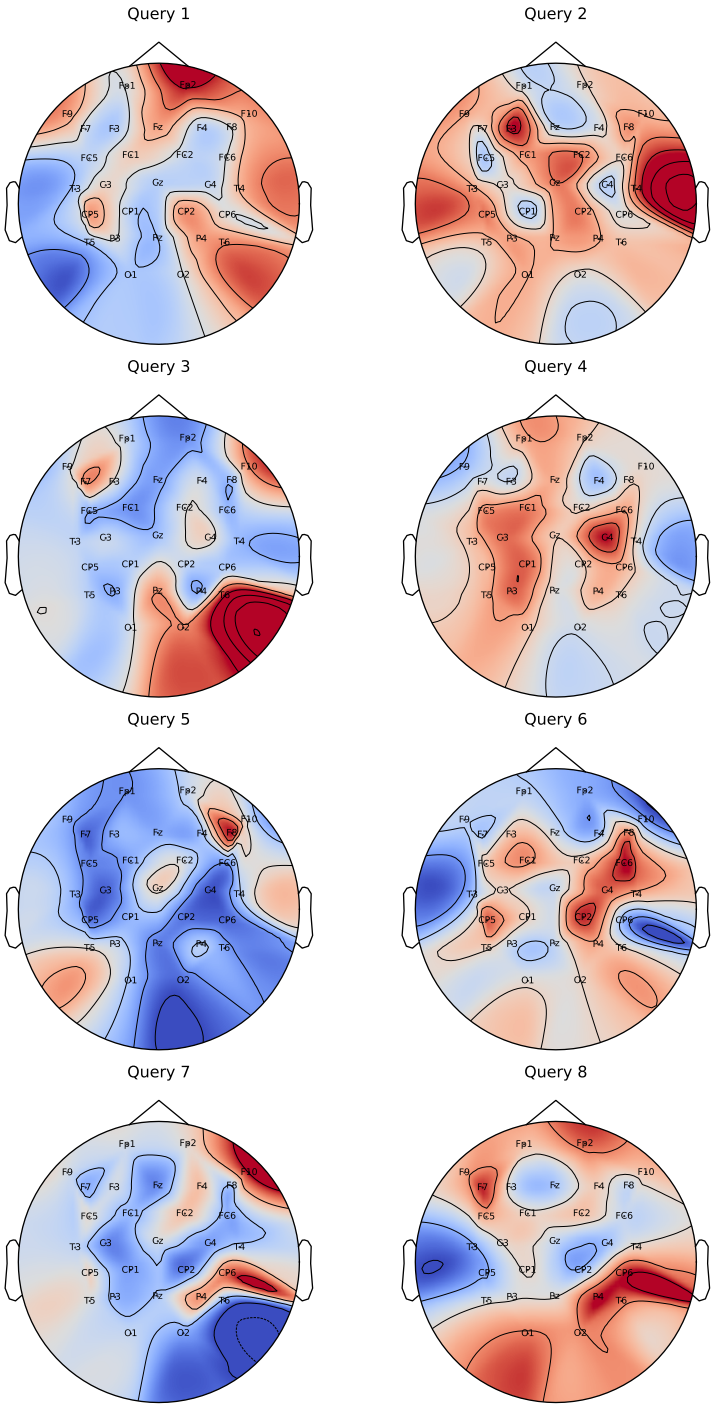


Figure 5.7.: Latent space learned by the queries.

5.5. Brain Connectivity Analysis

Visualization of the learned connectivity scores, presented in Figure 5.9, indicates patterns of inter-channel relationships learned by the model when trained on EEG data with different electrode topologies. The connectome plots present the cosine similarity of query representations, averaged over all queries, for each pair of EEG channels, where color intensity indicates the strength of connectivity. Only the strongest lines are shown in the plot.

The Siena dataset connectivity plot, recorded with a unipolar montage, shows inter-channel connectivity within each region, indicated by the local connections. There are also strong connections between the parietal area electrodes (CP5, CP1) and the electrodes in other regions, suggesting that the model also captures longer-distance connectivity patterns in the unipolar montage data. Similarly, the TUEG dataset's connectivity plot, which was recorded with a bipolar montage, shows both local and global connections. It also shows connections between channels that share a reference electrode, such as F8-T4 with C4-T4, which might be due to some correlations between these channels due to the shared electrode.

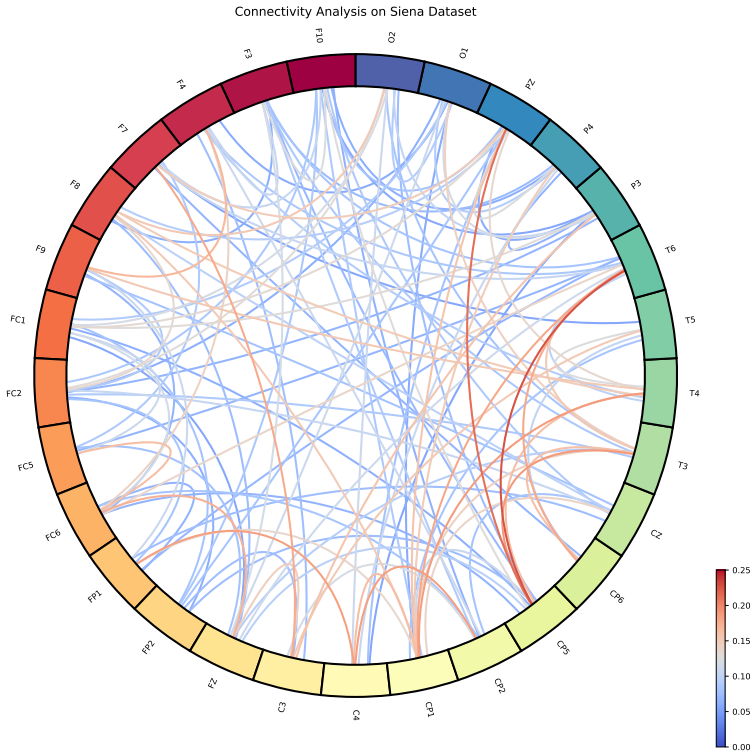


Figure 5.8.: Connectivity analysis on Siena dataset topology.

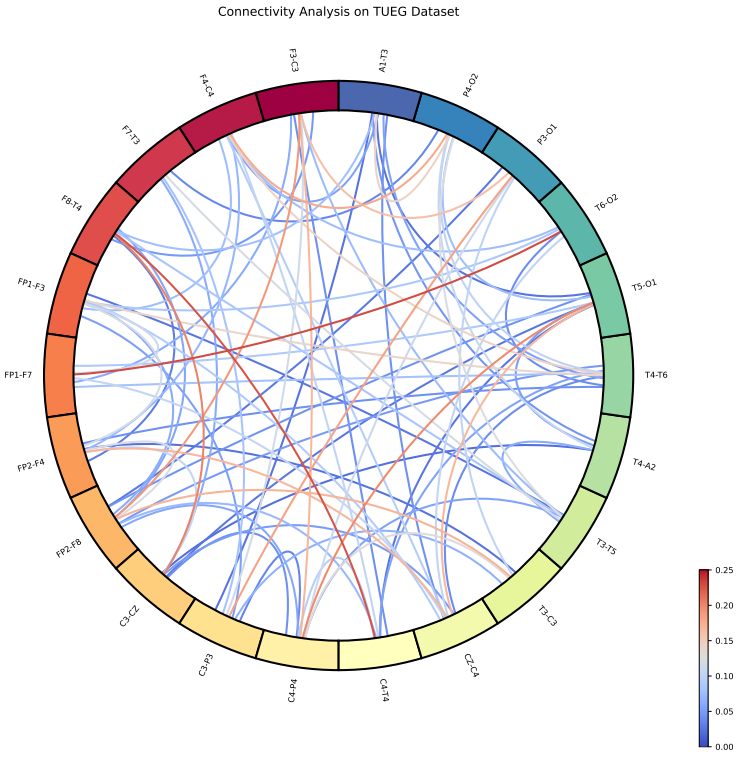


Figure 5.9.: Connectivity analysis on TUEG dataset topology.

5.6. Comparison to related work

The proposed model is compared to the state-of-the-art methods on the TUAB, TUAR, TUSL, and SEED-V datasets in Table 5.1, Table 5.2 and Table 5.3. The model shows comparable performance to the state-of-the-art methods on the TUAB dataset. The base model performs especially well compared to the models with similar sizes except for the base LaBraM model. Another interesting observation is that the model's performance does not scale with the model size, which is contradictory to the previous results, which show improvements with the bigger model sizes. This might show a problem with the pre-training setup of the large and huge models or might indicate the need for different hyperparameters for these models.

Table 5.1.: Performance Comparison on TUAB

Model	Model Size	Bal. Acc. (%)	AUPR	AUROC
Supervised Models				
SPaRCNet [52]	0.8M	78.96 ± 0.18	0.8414 ± 0.0018	0.8676 ± 0.0012
ContraWR [53]	1.6M	77.46 ± 0.41	0.8421 ± 0.0140	0.8456 ± 0.0074
CNN-Transformer [54]	3.2M	77.77 ± 0.22	0.8433 ± 0.0039	0.8461 ± 0.0013
FFCL [55]	2.4M	78.48 ± 0.38	0.8448 ± 0.0065	0.8569 ± 0.0051
ST-Transformer [56]	3.2M	79.66 ± 0.23	0.8521 ± 0.0026	0.8707 ± 0.0019
Self-supervised Models				
BENDR [35]	0.39M	76.96 ± 3.98	-	0.8397 ± 0.0344
BrainBERT [36]	43.2M	-	0.8460 ± 0.0030	0.8530 ± 0.0020
EEGFormer-Small [40]	1.9M	-	0.8620 ± 0.0050	0.8620 ± 0.0070
EEGFormer-Base [40]	2.3M	-	0.8670 ± 0.0020	0.8670 ± 0.0030
EEGFormer-Large [40]	3.2M	-	0.8720 ± 0.0010	0.8760 ± 0.0030
BIOT [38]	3.2M	79.59 ± 0.57	0.8692 ± 0.0023	0.8815 ± 0.0043
EEG2Rep [57]	-	80.52 ± 2.22	-	0.8843 ± 0.0309
FEMBA-Base [58]	47.7M	81.05 ± 0.14	0.8894 ±0.0050	0.8829 ± 0.0021
FEMBA-Large [58]	77.8M	81.47 ± 0.11	0.8992 ± 0.0007	0.8856 ± 0.0004
FEMBA-Huge [58]	386M	81.82 ± 0.16	0.9005 ± 0.0017	0.8921 ± 0.0042
CEReBrO [50]	3.58M	79.40± 0.19	0.8763 ± 0.0031	0.8749 ± 0.0033
CEReBrO [50]	39.95M	81.29 ± 0.15	0.8994 ± 0.0002	0.8867 ± 0.0006
CEReBrO [50]	85.15M	81.67 ± 0.23	0.9049 ± 0.0026	0.8916 ± 0.0038
LaBraM-Base [39]	5.8M	81.40 ± 0.19	0.8965 ± 0.0016	0.9022 ± 0.0009
LaBraM-Large [39]	46M	82.26 ± 0.15	0.9130 ± 0.0005	0.9127 ± 0.0005
LaBraM-Huge [39]	369M	82.58 ± 0.11	0.9204 ± 0.0011	0.9162 ± 0.0016
CBraMod [41]	68.4M	**82.49 ± 0.25**	**0.9221 ± 0.0015**	**0.9156 ± 0.0017**
LUNA-Base	7M	81.19 ± 0.41	0.8992 ± 0.0011	0.8906 ± 0.0002
LUNA-Large	43M	80.21 ± 0.52	0.8896 ± 0.0044	0.8823 ± 0.0043
LUNA-Huge	311.4M	80.91 ± 0.33	0.8978 ± 0.0016	0.8895 ± 0.0015

The proposed models are compared against other methods on TUAR and TUSL datasets in Table 5.2 below. All three different-sized models perform comparably, if not better, than the previous methods on these datasets. The base model performs better than all the previous methods on the TUAR dataset on the AUROC score. Compared to the observation in the TUAB dataset, the huge model shows improvements over the base model on these two datasets. For example, the huge model is the state-of-the-art model for the TUAR dataset on both AUROC and AUPR metrics, as well as on the AUROC metric on the TUSL dataset.

Table 5.2.: Performance Comparison across TUAR, TUSL

Model	Method Size	TUAR		TUSL	
		AUROC	AUPR	AUROC	AUPR
EEGNet [59]	-	0.752 ± 0.006	0.433 ± 0.025	0.635 ± 0.015	0.351 ± 0.006
TCN [60]	-	0.687 ± 0.011	0.408 ± 0.009	0.545 ± 0.009	0.344 ± 0.001
EEG-GNN [61]	-	0.837 ± 0.022	0.488 ± 0.015	0.721 ± 0.009	0.381 ± 0.004
GraphS4mer [62]	-	0.833 ± 0.006	0.461 ± 0.024	0.632 ± 0.017	0.359 ± 0.001
BrainBERT [36]	43.2M	0.753 ± 0.012	0.350 ± 0.014	0.588 ± 0.013	0.352 ± 0.003
EEGFormer-Small [40]	1.9M	0.847 ± 0.013	0.488 ± 0.012	0.683 ± 0.018	0.397 ± 0.011
EEGFormer-Base [40]	2.3M	0.847 ± 0.014	0.483 ± 0.026	0.713 ± 0.010	**0.393 ± 0.003**
EEGFormer-Large [40]	3.2M	0.852 ± 0.004	0.483 ± 0.014	0.679 ± 0.013	0.389 ± 0.003
FEMBA-Tiny [58]	7.8M	0.918 ± 0.003	0.518 ± 0.002	0.708 ± 0.005	0.277 ± 0.007
FEMBA-Base [58]	47.7M	0.900 ± 0.010	0.559 ± 0.002	0.731 ± 0.012	0.289 ± 0.009
FEMBA-Large [58]	77.8M	0.915 ± 0.003	0.521 ± 0.001	0.714 ± 0.007	0.282 ± 0.010
LUNA-Base	7M	0.923 ± 0.009	0.550 ± 0.032	0.683 ± 0.079	0.279 ± 0.011
LUNA-Large	43M	0.916 ± 0.011	0.532 ± 0.042	0.733 ± 0.052	0.274 ± 0.007
LUNA-Huge	311.4M	**0.924 ± 0.011**	**0.568 ± 0.042**	**0.748 ± 0.001**	0.283 ± 0.009

Finally, the models are compared against the state-of-the-art models on the raw features of the SEED-V dataset in Table 5.3. This dataset represents a significant challenge for the model as it has an entirely different electrode topology with 62 channels than the pre-training dataset consisting of Siena and TUEG. These results show the capability of the model to generalize into new topologies with only fine-tuning. As can be seen from the table, the model has significantly lower results on this dataset. The accuracy values are at least 4% lower, and the weighted F1 and Cohen's Kappa metrics are around 8 absolute points lower than the state-of-the-art models. This shows a current limitation of the model. The method is not able to generalize well into unseen topologies and requires future work to generalize beyond the pre-training data. One possible direction would be to improve the channel positional encoding approach, which is currently a learned embedding for each channel index. The set of positional encodings for the unseen channels is not trained during pre-training as the model never uses them. A good approach would be to dynamically form these channel positional encodings and not rely on the fixed set of embeddings. The comparison of the base, large and the huge models compared to the previous methods with similar sizes can also be seen in Figure 5.10, Figure 5.11 and Figure 5.12.

Table 5.3.: The results of different methods on emotion recognition (SEED-V, 5-classes).

Methods	Model Size	Balanced Accuracy	Cohen's Kappa	Weighted F1
SPaRCNet [52]	0.79M	0.2949 ± 0.0078	0.1121 ± 0.0139	0.2979 ± 0.0083
ContraWR [53]	1.6M	0.3546 ± 0.0105	0.1905 ± 0.0188	0.3544 ± 0.0121
CNN-Transformer [54]	3.2M	0.3678 ± 0.0078	0.2072 ± 0.0183	0.3642 ± 0.0088
FFCL [55]	2.4M	0.3641 ± 0.0092	0.2078 ± 0.0201	0.3645 ± 0.0132
ST-Transformer [56]	3.5M	0.3052 ± 0.0072	0.1083 ± 0.0121	0.2833 ± 0.0105
BIOT [38]	3.2M	0.3837 ± 0.0187	0.2261 ± 0.0262	0.3856 ± 0.0203
LaBraM-Base [39]	5.8M	0.3976 ± 0.0138	0.2386 ± 0.0209	0.3974 ± 0.0111
CBraMod [41]	7.3M	**0.4091 ± 0.0097**	**0.2569 ± 0.0151**	**0.4101 ± 0.0108**
LUNA-Base	7M	0.3552 ± 0.0114	0.1594 ± 0.0169	0.3139 ± 0.0180
LUNA-Large	43M	0.3363 ± 0.0113	0.1380 ± 0.0153	0.2955 ± 0.0127
LUNA-Huge	311.4M	0.3589 ± 0.0072	0.1616 ± 0.0085	0.3220 ± 0.0083

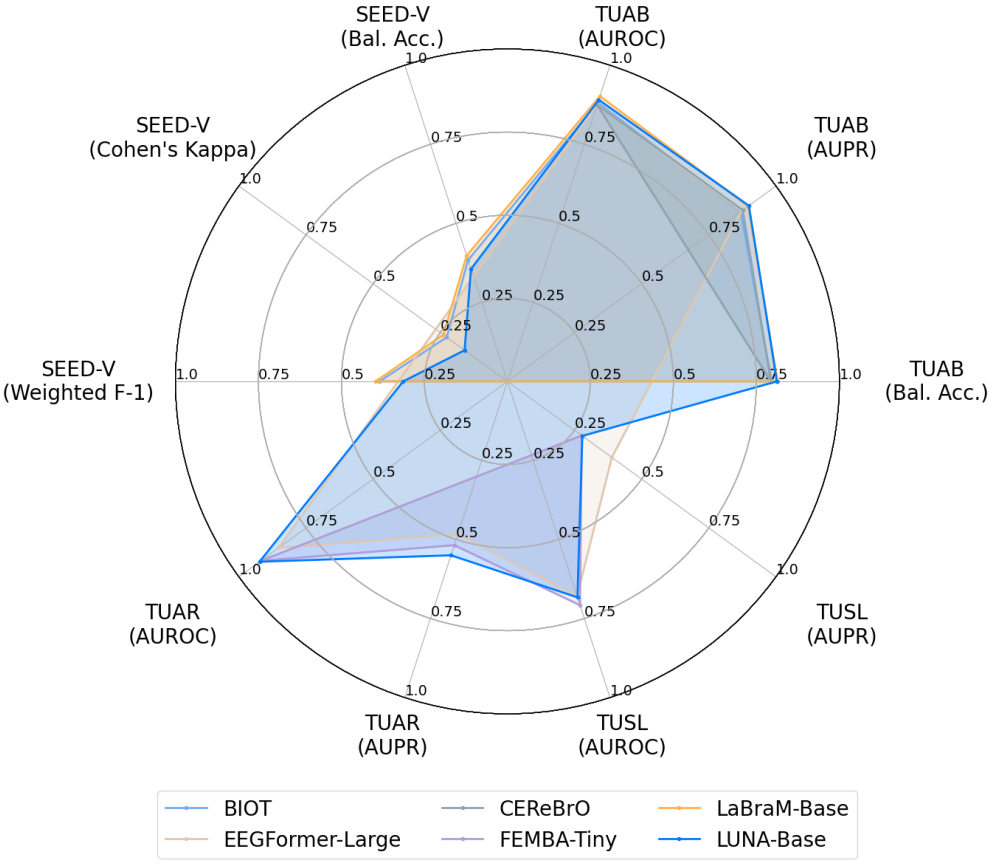


Figure 5.10.: Comparison of the base model with previous work with less than 10M parameters.

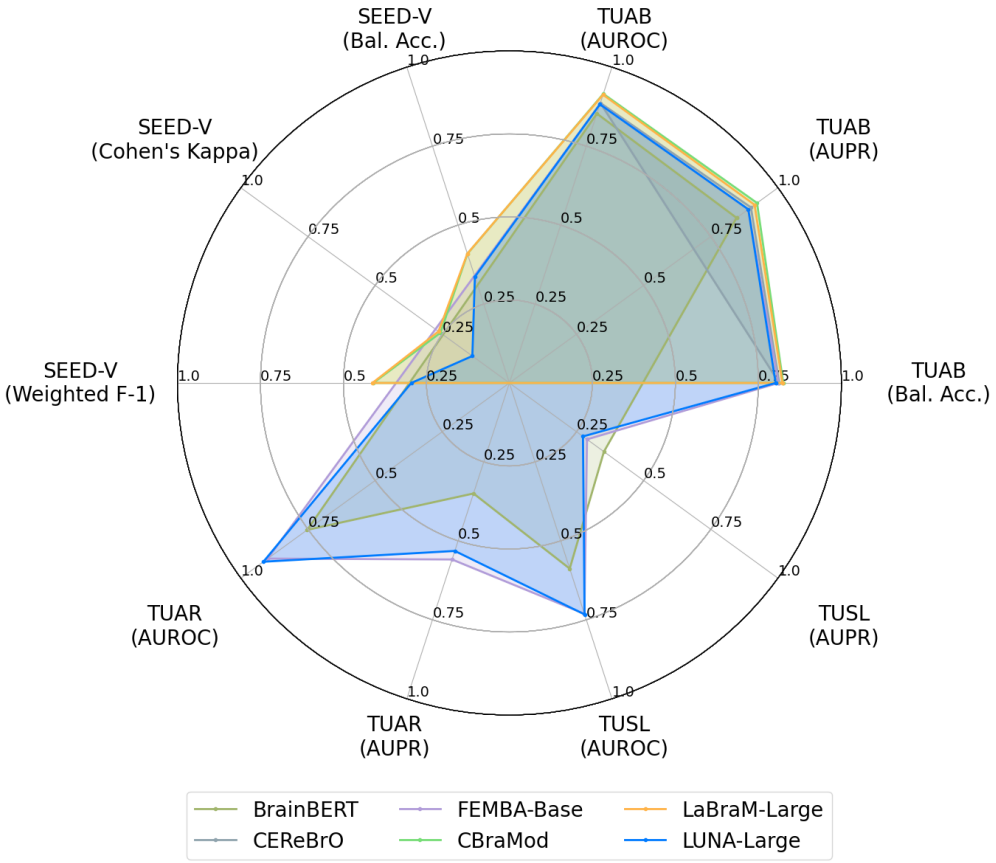


Figure 5.11.: Comparison of the large model with previous work with less than 70M parameters.

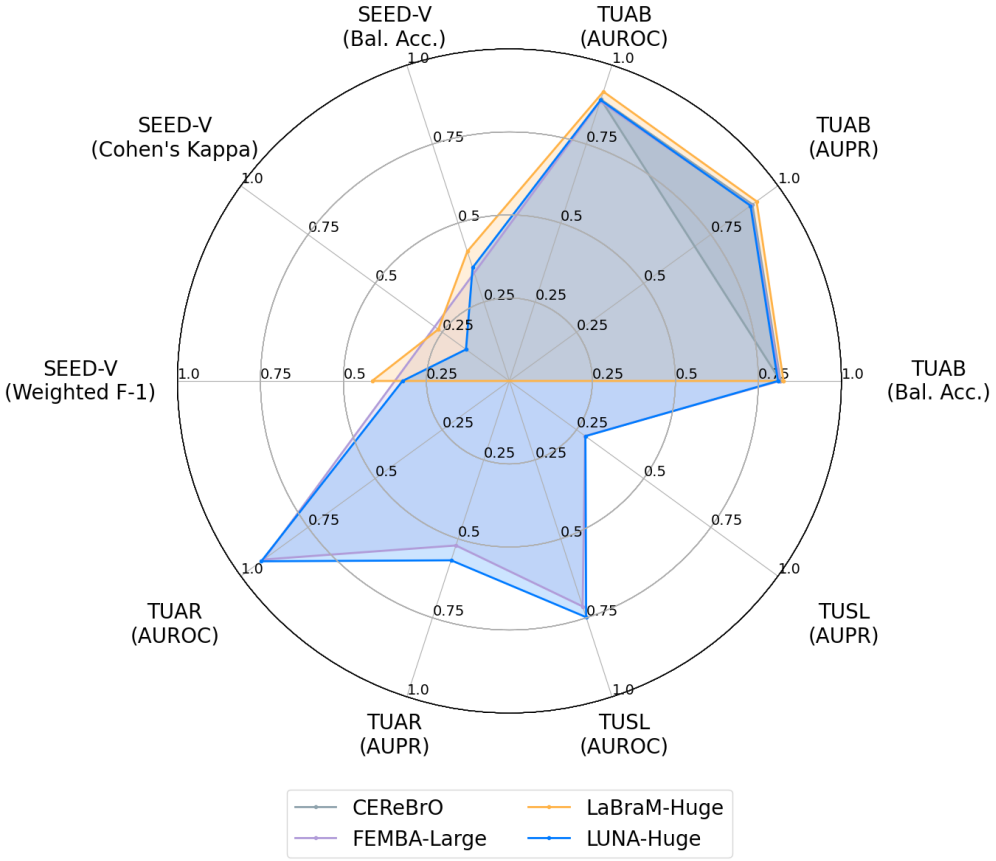


Figure 5.12.: Comparison of the huge model with previous work with more than 70M parameters.

5.7. Resource Requirement Analysis

As a final analysis, the complexity of the approach based on the number of floating operations and the memory usage is compared against the best previous methods in the literature: LaBraM [39] and CBraMod [41]. As discussed in the Chapter 4 section, the LaBraM method uses a full attention approach that applies attention to the flattened sequence of channels and patches, resulting in quadratic complexity on both the number of patches and the number of channels. CBraMod instead applies parallel attention to the patch and channel dimensions while merging the other dimension with the batch dimension. This method scales quadratically with the maximum of the number of channels or patches in the input. Compared to these methods, LUNA scales linearly with the channels and scales quadratically with the number of channels but has a lower overall complexity due to the smaller latent space size. Moreover, LUNA is trained with 16-bit precision, which further decreases its resource usage.

In Figure 5.13, the scaling of the methods is compared based on the number of patches while keeping the number of channels fixed at a realistic value of 22. The plot confirms the complexity analysis and shows a lower number of FLOPS and memory usage for the latent space approach compared to the full attention approach. The full attention approach uses all the available memory after only 300 patches, while LUNA can extend beyond 8000 patches. Furthermore, LUNA shows a lower scaling bound compared to CBraMod, confirming the lower theoretical bound $\mathcal{O}(B \cdot S^2 \cdot Q \cdot E)$ compared to CBraMod's $\mathcal{O}(B \cdot S^2 \cdot C \cdot E)$. Even though both approaches scale quadratically with the number of patches, the smaller latent size decreases the complexity of LUNA.

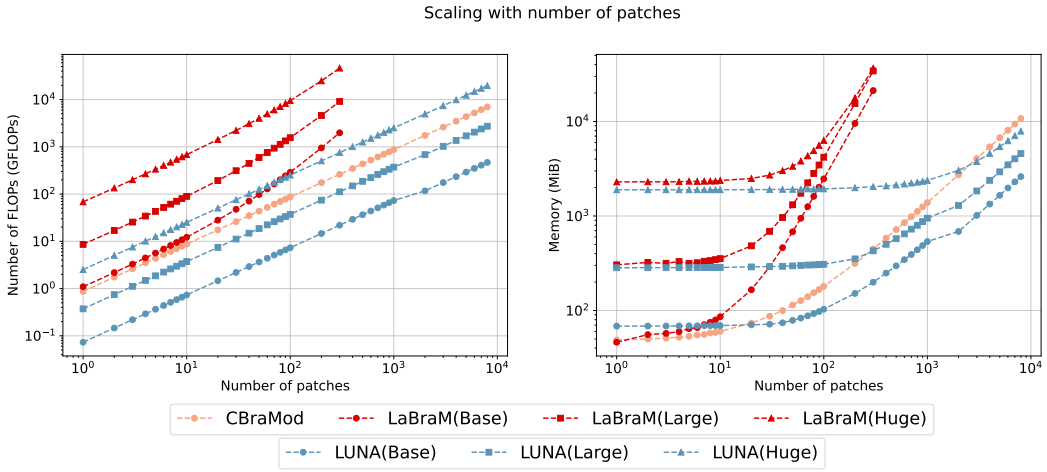


Figure 5.13.: Scaling of the number of FLOPS and the memory usage with the number of patches.

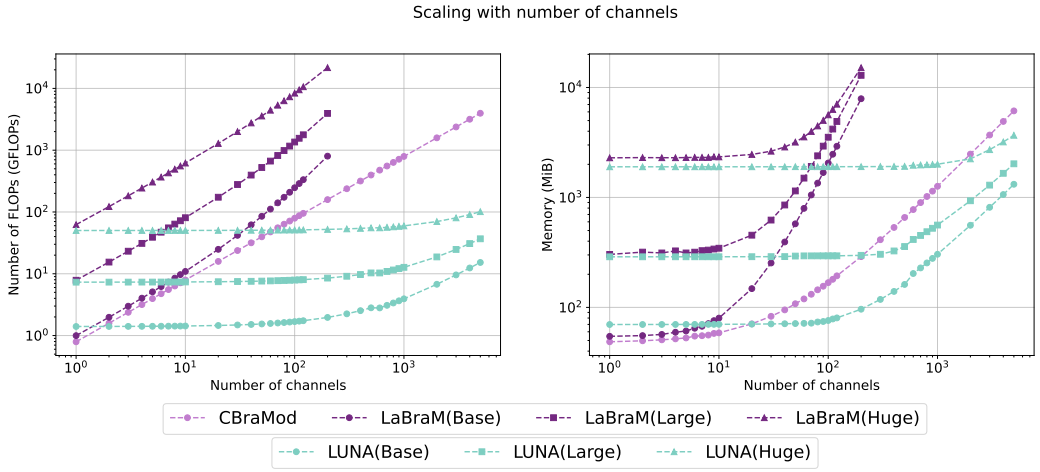


Figure 5.14.: Scaling of the number of FLOPS and the memory usage with the number of channels.

In Figure 5.14, the models are instead analyzed with the increasing number of channels while keeping the number of patches at 20. LaBraM approach performs similarly and shows a quadratic pattern. The CBraMod approach also shows a quadratic pattern after the number of channels becomes the complexity bottleneck instead of the number of patches. In contrast, LUNA shows a constant complexity represented by the fixed latent size for the number of channels ranging between 1 and 200. The cross-attention between the channels and the queries becomes the bottleneck only after this point, and the model shows increasing complexity. However, even after this point, the scaling is linear with the number of channels and much lower than other methods.

Both analyses show that the model is much more efficient than the state-of-the-art methods, showing a lower bound than the previous methods even with the huge model with a large number of patches or channels. These results show a promising direction for LUNA on very high-density scalp measurements with large amounts of channels or on very long sequences with a large number of patches.

5.8. Experiment Setup

The following tables show the hyperparameter setup for the pre-training and the downstream fine-tuning for LUNA.

Table 5.4.: Hyperparameters for EEG pre-training.

Hyperparameters		**LUNA-Base**	**LUNA-Large**	**LUNA-Huge**
Temporal Encoder	Input channels	{1,8,8}	{1,16,16}	{1,32,32}
	Output channels	{16,16,16}	{24,24,24}	{32,32,32}
	Kernel size		{20,3,3}	
	Stride		{10,1,1}	
	Padding		{9,1,1}	
Patch size			40	
Transformer encoder layers		8	10	24
Number of queries		4	6	8
Query size		64	96	128
Hidden size		256	576	1024
MLP size		1024	2304	4096
Attention head number		8	12	16
Batch size per GPU		2040	2040	720
Total batch size		8160	8160	11520
Peak learning rate			1.25e-4	
Minimal learning rate			2.5e-7	
Learning rate scheduler			Cosine	
Optimizer			AdamW	
Adam β			(0.9,0.98)	
Weight decay			0.05	
Total epochs			100	
Warmup epochs			10	
Loss type			Smooth-L1	
Non-masked region loss coefficient			0.05	
Query load balancing loss coefficient			0.2	
Gradient clipping			1	
Mask ratio			0.5	
Precision			bf16-mixed	

Table 5.5.: Hyperparameters for downstream fine-tuning.

Hyperparameters	**Values**
Batch size per GPU	512
Peak learning rate	1e-4
Minimal learning rate	5e-6
Learning rate scheduler	Cosine
Optimizer	AdamW
Adam β	(0.9,0.999)
Weight decay	0.05
Total epochs	50
Early stopping patience	10
Warmup epochs	5
Drop path	0.1 (B/L) 0.3 (H)
Layer-wise learning rate decay	0.5 (B) 0.8 (L/H)
Label smoothing (multi-class classification)	0.1 (B) 0 (L/H)

Chapter 6

Conclusion and Future Work

This thesis presented a novel EEG signal analysis method that uses a self-supervised learning approach to train topology invariant foundation models. The results and the analyses, detailed in the Results section, demonstrate the efficacy of the proposed method. Experiments on a variety of downstream tasks, including seizure detection and emotion recognition, validate that the model is comparable to or better than previous state-of-the-art solutions, even at greatly reduced memory and computational budgets. Therefore, empirical evidence supports that the query-based learned architecture effectively builds a topology-invariant representation and is able to overcome the issue of electrode configuration variability in EEG datasets successfully.

There are; however, also certain limitations of the approach, revealed through the results on the SEED-V dataset and the comparison between the models with different sizes. Specifically, while the model demonstrates strong generalization on the topologies observed during pre-training, its performance on the SEED-V emotion recognition dataset is not as good as on other tasks. The model exhibits lower performance compared to state-of-the-art methods on SEED-V, suggesting potential limitations in generalizing into unseen topologies. Moreover, the experiments with three different model sizes revealed that increasing the model size does not consistently yield significant performance improvements compared to the observations in the previous work, suggesting that the current architecture or pre-training strategy might have a problem or these models might require significantly different training setups than the smaller model.

Despite these limitations, this work presents a significant step towards robust and generalizable EEG analysis. Future work should focus on addressing the identified limitations and exploring the generalization to unseen topologies through changing the channel positional encoding and investigating scaling strategies to successfully train larger topology-invariant EEG models.

Appendix A

Task Description

Master Project at the Department of
Information Technology and Electrical Engineering

Autumn Semester 2024

Berkay Döner

EEG Signal Analysis with Foundation Models

September 17, 2024

Advisors: Thorir Mar Ingolfsson, thoriri@iis.ee.ethz.ch
Dr. Yawei Li, yawei.li@vision.ee.ethz.ch
Dr. Xiaying Wang, xiaywang@iis.ee.ethz.ch

Supervisor: Prof. Dr. Luca Benini, lbenini@iis.ee.ethz.ch

Handout: September 17, 2024
Due: March 07, 2025

Electroencephalography (EEG) is a non-invasive method used to record electrical activity in the brain, playing a critical role in both neurological research and clinical diagnostics [1]. By offering a window into the brain's activity, EEG helps diagnose and treat various neurological disorders. However, the analysis of EEG signals presents substantial challenges due to the complexity of these signals and the fine distinctions required between normal and abnormal brain activity [2].

In recent years, AI and deep learning have opened up new possibilities for analyzing EEG data. Foundation models—large, pre-trained neural networks that can be fine-tuned for specific tasks—have reshaped fields such as natural language processing and computer vision [3]. Their potential to improve EEG analysis is promising but still in its early stages.

Progress has been made in this area with models such as LaBraM, which segments EEG signals into channel patches to enable cross-dataset learning and fine-tuning [4]. Additionally, Brant-X offers a unified physiological signal alignment framework by leveraging EEG alongside other physiological signals, improving performance in EEG classification tasks like sleep stage and emotion recognition [5]. Another model, EEGFormer, takes a self-supervised learning approach with vector quantization to pre-train on large-scale EEG datasets, producing state-of-the-art results in tasks like seizure detection [6].

Despite these advancements, one major challenge in EEG analysis remains: handling EEG signals with varying numbers of channels. Tokenizing each EEG channel as an independent entity increases both computational complexity and memory demands, while ignoring important interactions between the channels [5, 6]. This project seeks to address these issues by proposing a method that uses cross-attention between learnable queries and the EEG signal's key/value pairs, unifying EEG signals across different datasets.

1 Project Description

The project aims to develop a generalized EEG foundation model that unifies EEG signals with varying channel counts. By applying cross-attention mechanisms, this approach will enable EEG signals to be transformed into a consistent format, preserving spatial relationships between channels while reducing computational complexity. This methodology makes it possible to efficiently tokenize EEG signals across all channels, addressing the drawbacks of previous approaches that treated each channel as an independent token.

The primary tasks for this project are as follows:

- **Task I: Literature Review**Review the latest advances in EEG foundation models, cross-attention mechanisms, and their applications in handling varying numbers of EEG channels. Study existing models like LaBraM, Brant-X, and EEGFormer to understand their methodologies and limitations.

- **Task II: Dataset Exploration and Preprocessing**
  Gather and preprocess publicly available EEG datasets, ensuring a variety of channel configurations. This task includes cleaning the data, dealing with noise and artifacts, and segmenting the signals in preparation for model training. The goal is to create a diverse, representative dataset that can be used for both pre-training and fine-tuning.
- **Task III: Cross-Attention Mechanism Development**
  Develop and implement a cross-attention mechanism between learnable queries and EEG signals. This approach will unify EEG signals from datasets with different channel configurations into a standardized format while preserving spatial dependencies between channels. Ensure the model's efficiency and scalability by reducing the tokenization complexity.
- **Task IV: Pre-training the Foundational Model**
  Investigate different pre-training techniques for EEG models, such as self-supervised learning and masked signal reconstruction. Implement the pre-training phase on large, unlabelled EEG datasets to learn universal representations of EEG signals. Key exploration areas include vector quantization, contrastive learning, and other representation learning strategies that can improve generalization across downstream tasks.
- **Task V: Fine-Tuning on Specific Tasks**
  Fine-tune the pre-trained foundational model on labeled EEG datasets for specific downstream tasks such as seizure detection, emotion recognition, and sleep stage classification. This task involves adjusting the pre-trained model to adapt to task-specific objectives, evaluating performance, and suggesting improvements.
- **Task VI: Evaluation**
  Conduct a comprehensive evaluation of the model's performance on various EEG datasets. Metrics such as accuracy, computational efficiency, and scalability will be assessed.
- **Task VII: Report and Presentation work** Work on the final report and final presentation for the group

2 Project Realization

2.1 Meetings

Weekly meetings and reports must be held. The exact time and location of these meetings will be determined within the first week of the project in order to fit the student's and the assistant's schedule. These meetings will be used to evaluate the status and progress of the project. Besides these regular meetings, additional meetings can be organized to address urgent issues as well.

2.2 Report

Documentation is an important and often overlooked aspect of engineering. One final report has to be completed within this project. The common language of engineering is de facto English. Therefore, the final report of the work is preferred to be written in English. Any form of word processing software is allowed for writing the reports, nevertheless, the use of LaTeX with Tgif[1] or any other vector drawing software (for block diagrams) is strongly encouraged by the IIS staff.

Final Report The final report has to be presented at the end of the project and a digital copy need to be handed in. Note that this task description is part of your report and has to be attached to your final report.

2.3 Presentation

There will be a presentation (15 min for the semester thesis, and 20 min for the MS thesis presentation followed by 5 min Q&A) at the end of this project in order to present your results to a wider audience. The exact date will be determined towards the end of the work.

References

[1] S. Weisdorf, S. W. Gangstad, J. Duun-Henriksen, K. S. S. Mosholt, and T. W. Kjær, "High similarity between eeg from subcutaneous and proximate scalp electrodes in patients with temporal lobe epilepsy," *Journal of Neurophysiology*, vol. 120, no. 3, pp. 1451–1460, 2018.

[2] T. M. Ingolfsson, S. Benatti, X. Wang, A. Bernini, P. Ducouret, P. Ryvlin, S. Beniczky, L. Benini, and A. Cossettini, "Minimizing artifact-induced false-alarms for seizure detection in wearable eeg devices with gradient-boosted tree classifiers," *Scientific Reports*, vol. 14, no. 1, p. 2980, 2024.

[3] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[4] W.-B. Jiang, L.-M. Zhao, and B.-L. Lu, "Large brain model for learning generic representations with tremendous eeg data in bci," *arXiv preprint arXiv:2405.18765*, 2024.

[1] See: `http://bourbon.usc.edu:8001/tgif/index.html` and `http://www.dz.ee.ethz.ch/en/information/how-to/drawing-schematics.html`.

[5] D. Zhang, Z. Yuan, J. Chen, K. Chen, and Y. Yang, "Brant-x: A unified physiological signal alignment framework," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 4155–4166.

[6] Y. Chen, K. Ren, K. Song, Y. Wang, Y. Wang, D. Li, and L. Qiu, "Eegformer: Towards transferable and interpretable large-scale eeg foundation model," *arXiv preprint arXiv:2401.10278*, 2024.

Zurich, March 6, 2025 Prof. Dr. Luca Benini

List of Figures

List of Tables

Bibliography

[1] M. Teplan *et al.*, "Fundamentals of eeg measurement," *Measurement science review*, vol. 2, no. 2, pp. 1–11, 2002.

[2] A. Craik, Y. He, and J. L. Contreras-Vidal, "Deep learning for electroencephalogram (eeg) classification tasks: a review," *Journal of neural engineering*, vol. 16, no. 3, p. 031001, 2019.

[3] O. Faust, Y. Hagiwara, T. J. Hong, O. S. Lih, and U. R. Acharya, "Deep learning for healthcare applications based on physiological signals: A review," *Computer methods and programs in biomedicine*, vol. 161, pp. 1–13, 2018.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[6] Y. Gong, Y.-A. Chung, and J. Glass, "Ast: Audio spectrogram transformer," *arXiv preprint arXiv:2104.01778*, 2021.

[7] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 2021, pp. 2114–2124.

[8] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, "Transformers in time series: A survey," *arXiv preprint arXiv:2202.07125*, 2022.

[9] L. Xu, M. Xu, Y. Ke, X. An, S. Liu, and D. Ming, "Cross-dataset variability problem in eeg decoding with deep learning," *Frontiers in human neuroscience*, vol. 14, p. 103, 2020.

[10] E. Kaniusas and E. Kaniusas, "Fundamentals of biosignals," *Biomedical Signals and Sensors I: Linking Physiological Phenomena and Biosignals*, pp. 1–26, 2012.

[11] F. J. Theis and A. Meyer-Bäse, *Biomedical signal analysis: Contemporary methods and applications*. MIT Press, 2010.

[12] G. H. Klem, "The ten-twenty electrode system of the international federation. the international federation of clinical neurophysiology," *Electroencephalogr. Clin. Neurophysiol. Suppl.*, vol. 52, pp. 3–6, 1999.

[13] M. A. Lopez-Gordo, D. Sanchez-Morillo, and F. P. Valle, "Dry eeg electrodes," *Sensors*, vol. 14, no. 7, pp. 12 847–12 870, 2014.

[14] K. E. Mathewson, T. J. Harrison, and S. A. Kizuk, "High and dry? comparing active dry eeg electrodes to active and passive wet electrodes," *Psychophysiology*, vol. 54, no. 1, pp. 74–82, 2017.

[15] J. N. Acharya and V. J. Acharya, "Overview of eeg montages and principles of localization," *Journal of Clinical Neurophysiology*, vol. 36, no. 5, pp. 325–329, 2019.

[16] R. Oostenveld and P. Praamstra, "The five percent electrode system for high-resolution eeg and erp measurements," *Clinical neurophysiology*, vol. 112, no. 4, pp. 713–719, 2001.

[17] H. Berger, "Über das elektroenkephalogramm des menschen," *Archiv für psychiatrie und nervenkrankheiten*, vol. 87, no. 1, pp. 527–570, 1929.

[18] P. Busia, A. Cossettini, T. M. Ingolfsson, S. Benatti, A. Burrello, V. J. B. Jung, M. Scherer, M. A. Scrugli, A. Bernini, P. Ducouret, P. Ryvlin, P. Meloni, and L. Benini, "Reducing false alarms in wearable seizure detection with eegformer: A compact transformer model for mcus," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 18, no. 3, pp. 608–621, 2024.

[19] A. Natarajan, Y. Chang, S. Mariani, A. Rahman, G. Boverman, S. Vij, and J. Rubin, "A wide and deep transformer neural network for 12-lead ecg classification," in *2020 Computing in Cardiology*. IEEE, 2020, pp. 1–4.

[20] M. Montazerin, E. Rahimian, F. Naderkhani, S. F. Atashzar, S. Yanushkevich, and A. Mohammadi, "Transformer-based hand gesture recognition from instantaneous to fused neural decomposition of high-density emg signals," *Scientific reports*, vol. 13, no. 1, p. 11000, 2023.

[21] Y. Song, Q. Zheng, B. Liu, and X. Gao, "Eeg conformer: Convolutional transformer for eeg decoding and visualization," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 31, pp. 710–719, 2022.

[22] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 3744–3753.

[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.

[24] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16 000–16 009.

[25] J. Dong, H. Wu, H. Zhang, L. Zhang, J. Wang, and M. Long, "Simmtm: A simple pre-training framework for masked time-series modeling," *Advances in Neural Information Processing Systems*, vol. 36, pp. 29 996–30 025, 2023.

[26] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[27] I. Obeid and J. Picone, "The temple university hospital eeg data corpus," *Frontiers in neuroscience*, vol. 10, p. 196, 2016.

[28] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals," *circulation*, vol. 101, no. 23, pp. e215–e220, 2000.

[29] P. Detti, "Siena scalp eeg database," *PhysioNet. doi*, vol. 10, p. 493, 2020.

[30] P. Detti, G. Vatti, and G. Zabalo Manrique de Lara, "Eeg synchronization analysis for seizure prediction: A study on data of noninvasive recordings," *Processes*, vol. 8, no. 7, p. 846, 2020.

[31] W. Liu, J.-L. Qiu, W.-L. Zheng, and B.-L. Lu, "Comparing recognition performance and robustness of multimodal deep learning models for multimodal emotion recognition," *IEEE Transactions on Cognitive and Developmental Systems*, 2021.

[32] W.-L. Zheng and B.-L. Lu, "Investigating critical frequency bands and channels for EEG-based emotion recognition with deep neural networks," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 3, pp. 162–175, 2015.

[33] R.-N. Duan, J.-Y. Zhu, and B.-L. Lu, "Differential entropy feature for EEG-based emotion classification," in *6th International IEEE/EMBS Conference on Neural Engineering (NER)*. IEEE, 2013, pp. 81–84.

[34] W. Zheng, W. Liu, Y. Lu, B. Lu, and A. Cichocki, "Emotionmeter: A multimodal framework for recognizing human emotions," *IEEE Transactions on Cybernetics*, pp. 1–13, 2018.

[35] D. Kostas, S. Aroca-Ouellette, and F. Rudzicz, "Bendr: Using transformers and a contrastive self-supervised learning task to learn from massive amounts of eeg data," *Frontiers in Human Neuroscience*, vol. 15, p. 653659, 2021.

[36] C. Wang, V. Subramaniam, A. U. Yaari, G. Kreiman, B. Katz, I. Cases, and A. Barbu, "Brainbert: Self-supervised representation learning for intracranial recordings," *arXiv preprint arXiv:2302.14367*, 2023.

[37] D. Zhang, Z. Yuan, Y. Yang, J. Chen, J. Wang, and Y. Li, "Brant: Foundation model for intracranial neural signal," *Advances in Neural Information Processing Systems*, vol. 36, pp. 26 304–26 321, 2023.

[38] C. Yang, M. Westover, and J. Sun, "Biot: Biosignal transformer for cross-data learning in the wild," *Advances in Neural Information Processing Systems*, vol. 36, pp. 78 240–78 260, 2023.

[39] W.-B. Jiang, L.-M. Zhao, and B.-L. Lu, "Large brain model for learning generic representations with tremendous eeg data in bci," *arXiv preprint arXiv:2405.18765*, 2024.

[40] Y. Chen, K. Ren, K. Song, Y. Wang, Y. Wang, D. Li, and L. Qiu, "Eegformer: Towards transferable and interpretable large-scale eeg foundation model," *arXiv preprint arXiv:2401.10278*, 2024.

[41] J. Wang, S. Zhao, Z. Luo, Y. Zhou, H. Jiang, S. Li, T. Li, and G. Pan, "Cbramod: A criss-cross brain foundation model for eeg decoding," *arXiv preprint arXiv:2412.07236*, 2024.

[42] K. Yi, Y. Wang, K. Ren, and D. Li, "Learning topology-agnostic eeg representations with geometry-aware modeling," *Advances in Neural Information Processing Systems*, vol. 36, pp. 53 875–53 891, 2023.

[43] G. Chau, C. Wang, S. Talukder, V. Subramaniam, S. Soedarmadji, Y. Yue, B. Katz, and A. Barbu, "Population transformer: Learning population-level representations of neural activity," *ArXiv*, pp. arXiv–2406, 2024.

[44] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer *et al.*, "Perceiver io: A general architecture for structured inputs & outputs," *arXiv preprint arXiv:2107.14795*, 2021.

[45] Y. Wu and K. He, "Group normalization," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[46] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[47] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *Neurocomputing*, vol. 568, p. 127063, 2024.

[48] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[49] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T. Liu, "On layer normalization in the transformer architecture," in *International conference on machine learning*. PMLR, 2020, pp. 10 524–10 533.

[50] A. Dimofte, G. A. Bucagu, T. M. Ingolfsson, X. Wang, A. Cossettini, L. Benini, and Y. Li, "Cerebro: Compact encoder for representations of brain oscillations using efficient alternating attention," *arXiv preprint arXiv:2501.10885*, 2025.

[51] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," *Journal of Machine Learning Research*, vol. 23, no. 120, pp. 1–39, 2022.

[52] J. Jing, W. Ge, S. Hong, M. B. Fernandes, Z. Lin, C. Yang, S. An, A. F. Struck, A. Herlopian, I. Karakis *et al.*, "Development of expert-level classification of seizures and rhythmic and periodic patterns during eeg interpretation," *Neurology*, vol. 100, no. 17, pp. e1750–e1762, 2023.

[53] C. Yang, D. Xiao, M. B. Westover, and J. Sun, "Self-supervised eeg representation learning for automatic sleep staging," *arXiv preprint arXiv:2110.15278*, 2021.

[54] W. Y. Peh, Y. Yao, and J. Dauwels, "Transformer convolutional neural networks for automated artifact detection in scalp eeg," in *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2022, pp. 3599–3602.

[55] H. Li, M. Ding, R. Zhang, and C. Xiu, "Motor imagery eeg classification algorithm based on cnn-lstm feature fusion network," *Biomedical signal processing and control*, vol. 72, p. 103342, 2022.

[56] Y. Song, X. Jia, L. Yang, and L. Xie, "Transformer-based spatial-temporal feature learning for eeg decoding," *arXiv preprint arXiv:2106.11170*, 2021.

[57] N. Mohammadi Foumani, G. Mackellar, S. Ghane, S. Irtza, N. Nguyen, and M. Salehi, "Eeg2rep: enhancing self-supervised eeg representation through informative masked inputs," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 5544–5555.

[58] A. Tegon, T. M. Ingolfsson, X. Wang, L. Benini, and Y. Li, "Femba: Efficient and scalable eeg analysis with a bidirectional mamba foundation model," *arXiv preprint arXiv:2502.06438*, 2025.

[59] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces," *Journal of neural engineering*, vol. 15, no. 5, p. 056013, 2018.

[60] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[61] S. Tang, J. A. Dunnmon, K. Saab, X. Zhang, Q. Huang, F. Dubost, D. L. Rubin, and C. Lee-Messer, "Self-supervised graph neural networks for improved electroencephalographic seizure analysis," *arXiv preprint arXiv:2104.08336*, 2021.

[62] S. Tang, J. A. Dunnmon, Q. Liangqiong, K. K. Saab, T. Baykaner, C. Lee-Messer, and D. L. Rubin, "Modeling multivariate biosignals with graph neural networks and structured state space models," in *Conference on health, inference, and learning*. PMLR, 2023, pp. 50–71.

[63] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.