Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Towards Semi-Supervised Region-Learning for Electroencephalography Models

Semester Project

Dustin Klebe, Jie-Ming Li, Lukas Wolf

{klebed, ljie, wolflu}@ethz.ch

Distributed Computing Group
Computer Engineering and Networks Laboratory
ETH Zürich

Supervisors:
Ard Kastrati
Prof. Dr. Roger Wattenhofer

July 18, 2023

Acknowledgements

We would like to express our sincere gratitude to Prof. Wattenhofer for providing us with the opportunity and resources to work on this project. We would also like to thank Prof. Langer from UZH for his constant support and for providing access to the datasets that were crucial in the project.

Furthermore, we are grateful to Ard Kastrati for his generous supervision and guidance throughout the project. Thank you for your trust in our ideas and approach and for leaving us the freedom of exploration.

Abstract

While DETRtime [1] is a powerful architecture for detecting events in time series signals, there are still several issues where it can be improved upon. Therefore, in this project report, we aim to investigate the state-of-the-art time series segmentation DETRtime model by addressing its three main downsides: overlapping boxes, model size, and generalization capabilities.

First, the nature of predicting potentially overlapping boxes seems unusual as a segmentation objective. Therefore, we argue that predicting change points may be the more natural approach.

Second, the large size and unscalable transformer architecture make DETRtime difficult to use in certain applications. We investigate the representations that DETRtime learns and explore alternative building blocks of the model.

Finally, while DETRtime has shown superior performance in multiple segmentation settings, its generalization capabilities need to be further investigated. Since the authors observed large performance differences in development and test sets, we propose a novel pretraining procedure that learns discrete box embeddings and reconstructs them, which allows us to learn better representations of EEG data streams based on the semantics of their subsequences.

Contents

Chapter 1

Introduction and Motivation

As the most efficient pathway for transmitting information to the brain, the human eye plays a pivotal role in our daily lives. Our ability to process visual stimuli and navigate our surroundings is key to our survival and success. As a result, eye gaze and visual information are frequently utilized as behavioral measures in cognitive science and psychology to investigate attentional focus, cognitive control, decision-making, and more [2]. Currently, eye gaze information is predominantly employed to detect variations in a participant's attention and adherence to a task.

Eye-tracking data is often inaccessible to many researchers due to the high cost of hardware and experimental setup. Recent studies have explored the use of machine learning techniques to compute eye gaze based on functional Magnetic Resonance Imaging (fMRI) signals and webcams, but these setups have their limitations in usability and cost. The fMRI signal, for instance, does not provide the temporal resolution required to track cognition at the level of eye gaze. Webcam-based methods, on the other hand, can be less expensive but come with usability issues.

As an alternative, EEG (Electroencephalography) is a widely used, cost-friendly, and safe method in cognitive neuroscience that allows the measurement of the brain's electrical activity over extended periods in clinical settings. EEG has been used to study a wide range of cognitive processes, including attention, perception, and memory. Therefore, it presents itself as a viable option for researchers interested in studying eye gaze in cognitive neuroscience. [3]

Given that many neuroscientific and psychological laboratories already possess the necessary hardware to collect EEG data, developing an eye-tracking approach that can predict eye gaze based on simultaneously measured EEG and electrooculography (EOG) data is a worthwhile goal. This approach could accelerate the scientific discovery of human behavior and neurological and psychiatric diseases. By combining EEG and EOG data, researchers can obtain a more comprehensive understanding of cognitive processes that involve eye movements, such as attention and decision-making. Furthermore, this approach can overcome the limitations associated with traditional eye-tracking methods and provide a

more accessible and cost-effective option for studying eye gaze in cognitive neuroscience. Therefore, developing a reliable EEG-based eye-tracking method has the potential to advance our understanding of the neural mechanisms underlying human behavior and psychiatric disorders. On the other hand, EEG data analysis is difficult due to its high dimensionality, noise, and complex structure.

The segmentation of time series signals is a fundamental problem in many fields, such as signal processing, machine learning, and data mining. In general, segmentation involves dividing a time series into meaningful segments that correspond to different underlying processes or phenomena. For instance, in speech recognition, the segmentation of audio signals into phonemes is necessary to identify words and phrases.

As such, a significant challenge in EEG analysis is segmenting EEG time series into meaningful segments that correspond to specific brain states, such as sleep stages or cognitive processes. This project builds upon our previous work on DETRtime, a segmentation model for EEG data. Segmenting EEG data is crucial because it allows for the identification of different brain states related to cognitive processes like attention, memory, or emotion. In sleep research, segmenting EEG signals into sleep stages is crucial for understanding the organization of sleep and its relationship with various physiological and psychological processes.

Segmentation of EEG data can also aid in detecting abnormalities or changes in brain activity that may be associated with neurological or psychiatric disorders. For example, epileptic seizures can be identified by segmenting EEG signals and recognizing abnormal patterns of brain activity that are indicative of seizures. Segmenting EEG signals can also facilitate the interpretation of intricate patterns of brain activity, such as those that occur during multi-tasking or decision-making. By segmenting EEG data into epochs that correspond to different cognitive processes, it becomes possible to investigate the neural mechanisms underlying these processes.

In addition, the segmentation of EEG data is essential for the development of EEG-based brain-computer interfaces, which allow individuals to control devices using their brain activity. By segmenting the EEG signals into different brain states, it is possible to map specific brain states to specific device commands.

The DETRtime architecture has proven its value within the domain of EEG time series segmentation, but one might expect that it generalizes beyond EEG to other medical time series signals or even time series segmentation in general. For instance, in financial analysis, the segmentation of stock prices into trends and patterns can help in predicting future stock prices. In environmental monitoring, the segmentation of time series data into different regimes can help in understanding the dynamics of ecological systems. This motivates a further investigation of the DETRtime model in time series segmentation, where we use EEG data as a starting point.

Chapter 2

Background and Related Work

2.1 Electroencephalography and Eye Tracking

EEG is a method that involves placing electrodes on the scalp to record the electrical activity in the brain. This technique is non-invasive and captures the combined activity of millions of neurons firing together, providing crucial insights into brain function and potential disorders.

In 1924, Hans Berger first recorded EEG by placing electrodes on his son's scalp and observing the electrical activity with a string galvanometer. He coined the term "electroencephalogram" for the recorded signals. Since then, EEG has become an essential tool in both clinical practice and neuroscience research [4]. In clinical settings, EEG is used to diagnose and monitor neurological disorders such as epilepsy, dementia, and sleep disorders. Meanwhile, in research, EEG is applied to investigate brain activity during cognitive and sensory processes such as memory, attention, and perception [5].

Eye tracking is a research technique that measures the movement and position of the eyes. It is a non-invasive method that records and analyzes the position of the pupil and the direction of the gaze. Eye tracking can provide valuable information about cognitive processes such as attention, perception, reading, and decision-making. The technique has numerous applications, including market research, usability testing, psychology, and neuroscience.

One of the earliest and most widely used methods of eye tracking is video-based eye tracking. This technique involves using a camera to record the position of the eyes and analyze the footage to determine the direction of the gaze. Infrared eye tracking is another popular method that uses infrared light to track the position of the pupils. More recently, mobile eye tracking has become increasingly popular, which allows researchers to study eye movements in naturalistic settings.

Eye tracking has been used in many different areas of research, including psychology, neuroscience, marketing, and human-computer interaction. For example, in psychology, eye tracking has been used to study attention, memory, and decision-making processes. In neuroscience, eye tracking has been used to study

the neural basis of visual attention and to develop new methods for diagnosing and treating neurological disorders.

Gaze prediction is a subject of active research that has applications in various fields such as human behavior analysis [6], advertisement [7], and human-computer interaction [8]. Previous research has found that attention facilitates action selection [9]. [10] demonstrated the possibility of performing activity recognition from eye movements. Some models use saliency maps to predict gaze location [11, 12], while others leverage machine learning techniques to estimate gaze position from indirect data. For example, [13] use webcam images, and [14] and [15] use functional magnetic resonance imaging (fMRI). Similarly, [16] reconstructed fixation maps directly from fMRI data to predict eye movement patterns. Recently, [17] found that it might be possible to infer gaze direction directly from EEG data. A big leap forward was being done by [3], which introduces a novel, large dataset and a benchmark with tasks of increasing difficulty.

Ocular events' detection is an active research topic with applications in human behaviour analysis [18], activity recognition [19], human-computer interaction and usability research [20], to mention a few.
The most standard and exploited segmentation techniques rely on infrared video-based systems. Modern eye-tracker companies use proprietary solutions to enable fast and reliable data segmentation [21]. Another prominent example of the segmentation framework was provided by [22]. This deep learning-based solution detects eye movements from eye-tracking data and does not require hand-crafted signal features or signal thresholding. However, in many experimental settings, a camera setup for eye tracking is not available, and thus this approach becomes impossible.
Another technique of measuring eye movements is Electrooculography (EOG), that record changes in electric potentials that originate from movements of the eye muscles [23]. Previous studies have described and evaluated algorithms for detecting saccades, fixations, and blinks characteristics from EOG signals recorded while driving a car [24]. The proposed algorithm detects microsleep episodes in eye movement data, showing the high importance of the tool. Other authors [25] have successfully evaluated algorithms for detecting three eye movement types from EOG signal achieving average precision of 76.1 % and recall of 70.5 % over all classes and participants. To date, the most successful and comprehensive approach to the problem of the EOG segmentation was proposed by [26]. They classified temporal eye parameters (saccades, blinks) from EOG data. The classification sensitivity for horizontal and large saccades was larger than 89% and for vertical saccades larger than 82%. Another line of research on the subject has been mostly restricted to blink detection [27]. Their BLINKER algorithms are effective in capturing a majority of the blinks and calculating common ocular indices.

Recently, [1] introduces a new way of segmenting EEG signals, achieving

state-of-the-art performance in ocular events and sleep stage segmentation. A more involved introduction to this work is provided in Section 2.3.1.

2.2 Machine Learning Background

Machine Learning, particularly deep learning, has become a popular technique for analyzing high-dimensional and complex data such as EEG data. Deep learning in particular allows extracting meaningful representations in a lower-dimensional space, thus reducing redundancy in the data while retaining key information. The resulting representations can be used for various downstream tasks such as time series segmentation, which is of particular interest in our research.

2.2.1 Representation Learning

Representation learning is a fundamental aspect of deep learning that is crucial for the performance of deep learning models. The goal of representation learning is to automatically learn a more compact and informative representation of the input data that can be used for various downstream tasks such as classification, segmentation, and synthesis. Representation learning operates under the premise of the manifold hypothesis [28] i.e. that a complex dataset, which exists in a high-dimensional space lies on a manifold that is parametrized by a lower-dimensional space. As such, we can learn representations of the data in this reduced parameter space.

Representation learning is performed by training deep neural networks with multiple layers of nonlinear transformations. Each layer of the network learns to transform the input data into a new representation that captures increasingly abstract and high-level features of the input data.

One of the key advantages of deep learning and representation learning is their ability to handle high-dimensional and complex data, such as images, audio, and in our case EEG. By learning a more informative representation of the input data, deep learning models can often achieve better performance on a wide range of tasks compared to traditional machine learning methods.

There are many different techniques and architectures for representation learning in deep learning, including convolutional neural networks (CNNs) for image and video data, recurrent neural networks (RNNs) for sequential data, and transformers for natural language processing. In addition, unsupervised learning methods such as autoencoders, generative adversarial networks (GANs), and variational autoencoders (VAEs) can also be used for representation learning without requiring labeled data.

Recently, Wav2Vec [29] introduced a model that can learn representations of high-dimensional continuous data in an unsupervised fashion, which can extract

high-quality representation of audio signal and transform it into text, i.e., a transformation from continuous signal into discrete. BENDR [30] extends this approach to EEG data with the aim of achieving high-quality representation of EEG data.

2.2.2 Time Series Segmentation

Time-series segmentation is a crucial task in machine learning that involves dividing a continuous sequence of data into characteristic segments or sub-sequences, each of which is assumed to exhibit a homogeneous behavior or pattern. This task is particularly important in applications where the data is naturally in the form of a time series, such as in finance, healthcare, and speech recognition. In the case of EEG data, segmentation can be used to identify different ocular events.

Time-series segmentation is an active field of research in machine learning, leading to a wide variety of methods, ranging from simple statistical methods to complex machine-learning algorithms. One common approach is to use unsupervised learning methods such as clustering [31], and dynamic time warping [32]. These methods aim to group together similar segments of the time series based on some distance or similarity measure. One prominent method is the hidden Markov [33] algorithm, which assumes that the time-series is the evidence of an underlying generative procedure. The method tries to learn the generative function for each class explicitly, while maximizing the likelihood of the evidence.

Another approach to time-series segmentation is to use supervised learning methods, where the goal is to learn a mapping between the input time series and a set of labeled change-points, which indicate the segment boundaries. This approach requires labeled training data and can be formulated as a classification and regression problem, as the position and the class of the change-point are predicted.

Deep learning models have also been applied to time-series segmentation tasks, with great success. One common approach is to use recurrent neural networks (RNNs [34]) or convolutional neural networks (CNNs) to learn a hierarchical representation of the time series and then use this representation to perform segmentation. Other deep learning models, such as transformers [35], have also been applied to time-series segmentation tasks with promising results.

One of the key challenges in time-series segmentation is the trade-off between accuracy and computational efficiency. While complex machine learning models can achieve high accuracy, they are often computationally expensive and require large amounts of training data. On the other hand, simpler methods such as statistical models may be faster and require less data, but may not be as accurate.

2.2.3 Region Learning

We define *Region learning* as the intersection of representation learning and time series segmentation. The goal of region learning is identifying and learning semantically meaningful fixed-sized representations of variable length segments within a time series. Therefore, regions are represented in a way that enables easy segmentation and prediction in downstream tasks. The problem is motivated by the insight that complex signals like EEG can be divided into multiple regions events (e.g. ocular events or sleep stages), where each event follows its own generating distribution.

To elaborate further, region learning involves identifying segments in time series data that are meaningful and representative of underlying patterns. These segments could be contiguous time windows, specific events or patterns within the time series, or other forms of meaningful sub-sequences. Once identified, these regions can be represented using appropriate machine learning techniques, such as feature extraction, encoding, or embedding, to capture relevant information for downstream tasks.

One example of a downstream task that could benefit from region learning is time series classification, where the goal is to predict the class or category of a time series based on its features. By using region learning to identify and represent meaningful segments within the time series, the task of classification can become more accurate and efficient, as the algorithm can focus on the most informative parts of the time series.

Region learning is closely related to concepts computer vision, as both fields deal with learning representations of semantically meaningful segments. In computer vision, this involves identifying and segmenting objects, scenes, or other meaningful units within an image or video. Region learning in time series data is analogous to this process, as it involves identifying and segmenting meaningful units within a temporal sequence. However, region learning in time series data presents unique challenges, such as dealing with irregular or non-uniform time intervals, varying sampling rates, and temporal dependencies between segments. These challenges require specialized techniques and algorithms that are tailored to the specific characteristics of time series data.

2.3 Considered Models

2.3.1 DETRtime

Architecture and Design

DETRtime [1] is a transformer-based architecture that was developed as a modification of the original DETR (DEtection TRansformer) [36] model, which was

used for object detection in natural images. However, DETRtime is specifically designed for detecting events in a time series and has been extensively tested on EEG data. The task of DETRtime is to segment a stream of EEG data into ocular events, which are events related to eye movements.

The architecture of DETRtime consists of two main components: a backbone and an encoder-decoder transformer. The backbone is a convolutional neural network (CNN) architecture that is composed of six layers of convolutional modules with residual connections that maps the input to a shape suitable for the transformer. The transformer, on the other hand, consists of an encoder and a decoder. The transformer encoder learns a sequence representation which, along with learned queries, is fed to the decoder to produce encoded features for a fixed number of learned event queries. DETRtime uses N=20 queries for almost all EEG data streams of length 1 second. The decoder attends to the encoder output and predicts the properties of each event.

The authors of DETRtime performed extensive experiments on their novel and publicly available datasets, which included the Visual Symbol Search, Reading paradigms, and Dots paradigms. According to the reported results the authors observed that their model outperformed the current state-of-the-art models by a large margin. Additionally, DETRtime was evaluated in different applications, such as the sleep staging task, which was initially performed with the U-time and SalientSleepNet models. In this task as well, DETRtime outperformed the current state-of-the-art solutions, demonstrating its generalization capabilities.

Overall, DETRtime is a powerful architecture that is capable of detecting events in a time series, with a focus on EEG data. Its backbone and transformer components work together to efficiently encode and decode features from the input data, and its superior performance in experiments makes it a promising solution for various applications in the future.

Limitations

While DETRtime has shown to be a powerful architecture for detecting events in a time series, there are some downsides that call for further investigation. One such downside is the nature of overlapping boxes, which is unusual as a segmentation objective. An alternative and potentially promising objective would be using change point detection. This approach is covered in Section 3.2.

Another issue is a large size and unscalable transformer architecture of DETRtime, which may limit its practical applications. This may call for further research into ways to optimize the architecture to make it more scalable and efficient. In the following chapters (see Section 3)

Moreover, while DETRtime has demonstrated superior performance in exper-

iments, its lack of generalization with large differences in development and test sets is a concern. It is possible that the model was overfitting to the specific datasets used in the experiments, and thus it may not perform as well on other EEG datasets or in different applications. Therefore, further investigation and testing on diverse datasets are necessary to evaluate the generalization capabilities of the model. In order to do so, we investigate learning better representations of EEG data in Section 4.

2.3.2 BENDR

Architecture and Design

BENDR [30] tackles the challenge of using deep neural networks (DNNs) to classify raw electroencephalography (EEG) data in brain-computer interface (BCI) applications. The goal is extracting useful features from raw sequences and classifying those features from limited data. The authors argues that self-supervised sequence learning, specifically techniques inspired by language modelling, could be an effective approach for developing more complex DNNs in BCI. The authors evaluate a self-supervised speech recognition technique called wav2vec 2.0 [29] adapted to EEG data, and discuss the transferability of learned features to unseen subjects, hardware, and tasks.

The approach involves using self-supervised training objective to learn compressed representations of raw data signals, which can be fine-tuned to a variety of downstream BCI and EEG classification tasks. The study found that a single pre-trained model can model completely novel raw EEG sequences recorded with differing hardware, and different subjects performing different tasks and outperforms prior work in more task-specific self-supervision.

Unlike common unsupervised schemes, BENDR uses a cosine similarity-based contrastive loss [37] to learn meaningful representations:

$$\mathcal{L} = -\log \frac{exp(cossim(c_t, b_t)/\kappa)}{\sum_{b_i \in B_D} exp(cossim(c_t, b_i)/\kappa)} \tag{2.1}$$

In essence, this loss operates by adjusting the output of the transformer at position t to be most similar to the encoded representation at t, as compared to a set of randomly sampled negative examples B_D, despite that this input to the transformer is masked. A more detailed explanation of this objective can be found in Section 4.2.1.

Limitations

The BENDR architecture has some limitations that need to be addressed. The first problem is related to the heavy downsampling applied by the model, which results in the loss of fine-grained local information. This information might be crucial for some tasks, such as the segmentation of ocular events, where saccades and blinks can last only 60 milliseconds or even shorter. Moreover, the heavy downsampling requires a long input sequence of 60 seconds, which is not practical for many tasks. Shorter input sequences, such as 1 second, would result in intermediate sequence lengths of less than 7 and hence, most of the sequential information would be lost.

Another limitation of BENDR is that it is restricted to downstream classification tasks, where the effective sampling rate is not critical. However, in other downstream tasks, it may be more important. Therefore, it is necessary to demonstrate the performance of the model on tasks other than classification. Overall, these limitations highlight the need for further research in developing architectures that can handle fine-grained local information while still maintaining computational efficiency and practicality for various applications.

2.4 Data Sources

2.4.1 EEGEyeNet

The dataset utilized in this study was originally published in [3], consisting of recordings from 365 healthy adults, with 190 females and 175 males, ranging in age from 18 to 80 years. The experiments were conducted in a soundproof and darkened room, with participants seated at a distance of 68cm from a 24-inch ASUS ROG Swift PG248Q monitor, featuring display dimensions of 531×299 mm and a resolution of 800×600 pixels, resulting in a display area of 400×298.9 mm and a vertical refresh rate of 100Hz. Specifically, data were collected from a subset of 80 healthy adults, including 33 females, with ages ranging from 20 to 40 years, as detailed in [1].

The EEG data was captured using a 128-channel EEG Geodesic Hydrocel system, with midline central recording reference and a sampling rate of 500 Hz. Simultaneously, an infrared video-based ET EyeLink 1000 Plus was used to record eye position at the same sampling rate of 500 Hz, with a spatial resolution of less than 0.01 root mean square (RMS) between successive samples. Participants were positioned 65 cm away from a 24-inch monitor with a resolution of 800 x 600 pixels, and the setup is illustrated in Figure 2.1. Ground truth signal was derived from the eye movement recordings [3].

EEG data can be easily corrupted by environmental factors such as temperature, humidity, and electromagnetic radiation, as well as by individual-specific

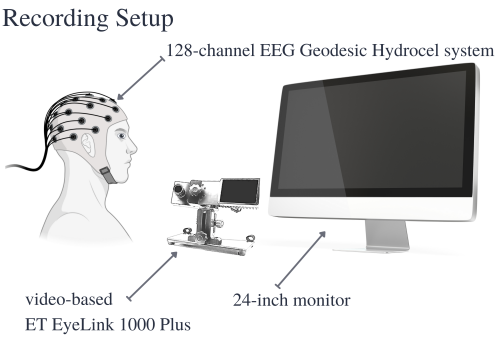

Figure 2.1: The illustration of the recording setup

disturbances such as eye blinks, muscle noise, and heart signal. To address this issue, the data used in this study was preprocessed through artifact correction [38]. Two preprocessing methods are provided by the Pedroni et al. toolbox: minimal and maximal preprocessing. The minimal approach involves detecting and interpolating malfunctioning electrodes, and filtering the data with a 40 Hz high-pass filter and a 0.5 Hz low-pass filter. In contrast, the maximal preprocessing method removes a larger number of artifacts by utilizing independent component analysis (ICA) in conjunction with *IClabel* [39], a pre-trained classifier that estimates the probability of a component representing artifactual activity. If the probability estimation of a component exceeds 0.8 for any artifact class, it is removed from the data. Importantly, minimally preprocessed data retains ocular artifacts, which is expected to facilitate gaze position estimation [3].

Large Grid

The experiment involves participants fixating on a series of dots presented in a sequence of 27 trials per block. The dots are displayed at 25 different screen positions, with the center dot appearing three times. Each dot is shown for a duration of 1.5 to 1.8 seconds. The positions of the dots were chosen to cover all areas of the screen, including the corners and center. The grid used for eye gaze estimation is based on the Large Grid paradigm developed by Son et al. (2008) for fMRI studies. The stimulus length and number of repetitions have been adjusted for our study, with different pseudo-randomized orderings of dot presentation used in five experimental blocks to increase the number of trials and reduce predictability. An illustration can be found in [3]

Processing Speed Paradigm

The Visual Symbol Search (VSS) is an assessment tool that measures processing speed in a computerized version of the Symbol Search Subtest of the Wechsler

Intelligence Scale for Children IV (WISC-IV) and the Wechsler Adult Intelligence Scale (WAIS-III). During the VSS test, participants view 15 rows at a time, where each row contains two target symbols, five search symbols, and two additional symbols with the words "YES" and "NO". The goal is to click the "YES" or "NO" symbol to indicate whether one of the target symbols is among the search symbols. Each VSS recording lasts for 120 seconds with a maximum of 60 trials, or rows. Half of the trials include one of the target symbols, and the other half do not. After each set of 15 rows, participants click a "next page" button to proceed to a new set of 15 rows. Participants are instructed to complete as many trials as possible within the given 120 seconds. Prior to the recording, participants perform a training of four trials with feedback. The collected data can be used to investigate the correlation between processing speed and behavior and neurophysiology.

ZuCo 2.0

Another dataset being used is called ZuCo 2.0, which is a collection of simultaneous eye-tracking and electroencephalography (EEG) data during natural reading and annotation. It contains data from 18 participants and includes 739 English sentences, with 349 being in a normal reading paradigm and 390 in a task-specific paradigm where participants actively search for a semantic relation type in the given sentences as a linguistic annotation task. [40]

2.4.2 Movie Watching Paradigm

In addition to the previously mentioned experimental paradigms from EEGEyeNet, we augment the dataset with the *Movie Watching* paradigm, representing a more natural setting. Data were obtained during two short and highly engaging movies scenes (‘Despicable Me’ [171 seconds clip, MPEG-4 movie, the bedtime ("Three Little Kittens") scene] and ‘Fun Fractals’ [163 seconds clip, MPEG-4 movie]). The data was first released in [1].

2.4.3 Data Annotation and Preprocessing

Literature studying eye movement classifies three different events, namely saccades, fixations and blinks [41]. A saccade is a time interval of very fast eye movement that instantly changes the position of the eye gaze. Fixations are generally defined as time periods without saccades, whereas blinks can be seen as a special kind of fixation, where the measured pupil diameter is equal to zero. While most of the important information is contained in the saccades, the time intervals of the fixations seem to contain artifacts of the saccades they follow.

In EEGEyeNet [3] the authors utilize various paradigms and datasets. Specifically, the paper processes fixed 500-length samples of data from three different paradigms, namely dots (Large Grid), processing speed, and ZuCo 2.0 (reading). We follow the preprocessing steps in [3]. In addition to the above paradigms, we also utilize movie data, which provides a more natural paradigm for analysis. For the DETRtime investigation (see Section 3), we use segmentation samples of length 500. The events in these samples are classified into three categories: fixations, saccades, and blinks. In order to pre-train models (see Section 4), we use the whole data stream collected during an experiment, instead of short fixed-length samples. This results in much more data being used for pre-training. The data is randomly sampled from all paradigms, while each participant is only included in either the train, validation or test split. For each epoch, the random offsets of the stream samples of arbitrary but fixed length (e.g. 2000) are resampled, leading to more variance in the training data. This approach helps to avoid overfitting and improves the generalization of the model.

2.4.4 Comparison to BENDR Dataset and Preprocessing

The BENDR paper [30] uses a combination of publicly accessible EEG data classification tasks and the Temple University Hospital EEG Corpus (TUEG) dataset for pre-training. The TUEG dataset consists of clinical recordings of over 10,000 people, recorded over many sessions and distributed across large time scales. The publicly accessible EEG data classification tasks include BCI task datasets and one sleep stage classification (SSC) task dataset. The BCI task datasets are classified in the context of particular trials or events, while the SSC dataset requires labeling of large spans of time with the particular sleep stage a subject is undergoing. The sequences in the SSC dataset are longer and closer in length to the pre-training task, allowing for consideration of how effective the approach is at a different time scale. The SSC dataset has a larger scale of available labels, enabling prior work to consider deeper and more complex models. The sequences in the datasets are segmented into 30-second periods and focused on 5 labels. [30]

The preprocessing steps of BENDR involve modifying downstream datasets to match the configuration of the pre-training dataset. The steps include removing spurious differences in channel amplitude by linearly scaling and shifting each sequence, adding a single channel to account for lost relative amplitude information, and addressing differences in sampling frequency and electrode sets using standard features in DN3. A reduced subset of the Deep1010 channel mapping was used, and sequences of 60 seconds were extracted for pre-training. Downstream datasets used shorter sequence lengths, but the architecture used was agnostic to the sequence length. [30]

In our pretraining experiments (see Section 4) we adopt the normalization

procedure of BENDR, and add the additional channel to encode the amplitude. For the dataset being used in the pre-training procedure of BENDR large-scale training infrastructure is needed. In order to extend the BENDR results from classification-only to other tasks, we pre-train our own BENDR model on the EEGEyeNet + movie paradigm dataset. This amounts to roughly 205GB of EEG data and contains a large variety of eye movements.

Chapter 3

Investigating DETRtime

3.1 Investigating DETRtime Backbone Embeddings

The goal of this section is to gain insights into the representation learning process of the backbone architecture of the DETRtime model [1]. As the dimensionality of the input data is not temporally downsampled, there is a one-to-one correspondence between the input timestamp and the intermediate representation output of the backbones. The similarity of timestamps within and across classes is analyzed using cosine similarity as a metric.

3.1.1 Raw EEG Timestamp Similarity

Analysis of the raw EEG input timestamps (see image 3.1) reveals that there is no correlation between timestamps across different classes, indicating that the vectors are orthogonal. Additionally, there is only a slight correlation between timestamps within each class, with blinks being the only exception, showing a higher within class-correlation than fixations and saccades. These results suggest that blinks may be easier to detect than saccades, even though they are even more underrepresented in the dataset than saccades.

3.1.2 Intermediate Timestamp Representation Similarity

Analysis of the intermediate timestamp representation after passing the data through the DETRtime backbone reveals that the cosine similarity values are clearly positive for all cross- and in-class pairs. Moreover, the similarity of timestamps within each class is higher than the cross-class similarity, indicating that the backbone has learned relevant features characterizing each class.

It is noteworthy that the similarity values of saccades are significantly larger than those of other classes, even though the similarity values of saccades on the raw timestamps are the smallest. This indicates that the backbone architecture specifically focuses on identifying informative features for the saccade class.

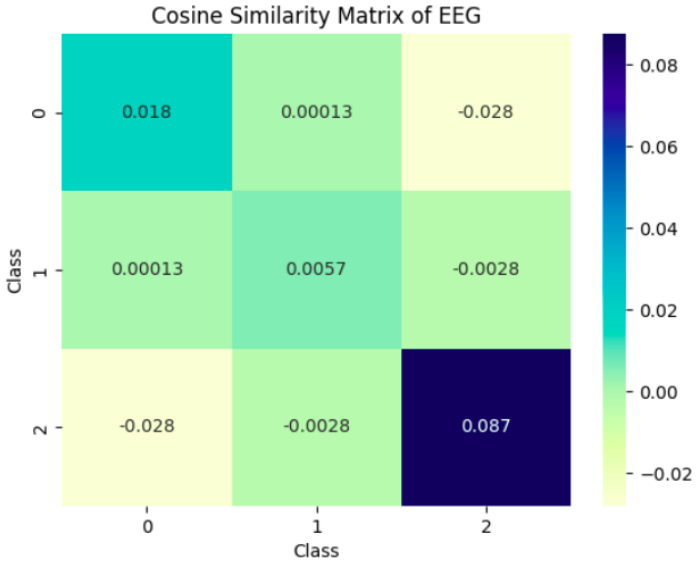

Figure 3.1: Cosine Similarity on Raw EEG Timestamps

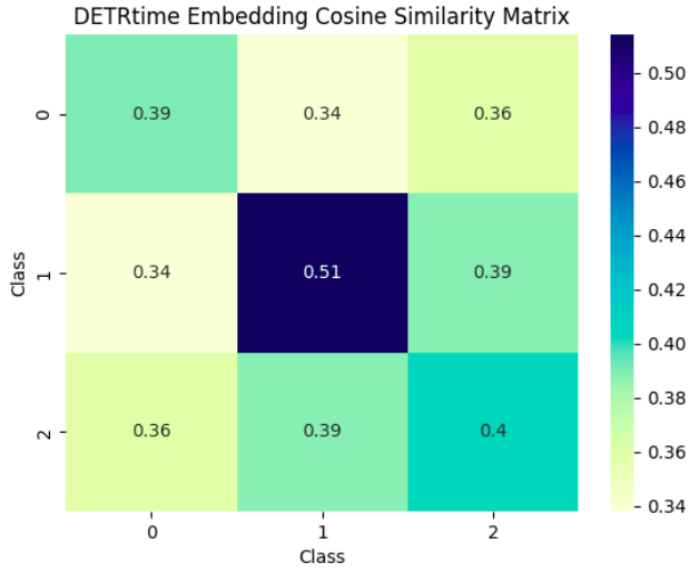

Figure 3.2: Cosine Similarity on DETRtime embedding

Thus, it is suggested that saccades may be difficult to classify and require specific consideration by the backbone.

3.1.3 Distilling the Backbone Through Pretraining

The analysis of the intermediate timestamp representation in DETRtime provides valuable insights into the representation learning process of the backbone architecture. The results indicate that the backbone has learned relevant features that characterize each class.

Separation of Classes

However, the fact that saccades have the highest similarity values on the intermediate timestamp representation, even though they have the smallest similarity values on the raw timestamps, suggests that there is a discrepancy between the learned features and the actual informative features. This discrepancy could potentially be addressed by investigating the embeddings and trying to pretrain our own embeddings with a different training objective. By doing so, we may be able to get rid of the backbone architecture or try to improve DETRtime by using these pretrained embeddings. Additionally, the fact that blinks show a clear correlation of similar timestamps suggests that blinks may be easier to detect than saccades, even though they are even more underrepresented in the dataset. Therefore, further investigation into the embeddings could potentially improve the overall performance of the model in detecting saccades and other underrepresented classes.

Cosine Similarity Approach

Since the backbone clearly tries to align same class time stamps while being trained on the DETRtime training objective, we now want to find out how much we can separate the classes with respect to cosine similarity, if cosine similarity was used as training objective.

In order to do so, we map single-timestamp inputs with either a 3-layer Perceptron or a multi-layer CNN to a lower dimensional vector. Across a batch, we then minimize the cosine-similarity across vectors of different classes, and maximize the similarity across same classes. The results of this learning procedure can be found in Figure 3.3.

As we can see, the cosine similarity within the classes fixation and blink is very high. Taking into account that most timesteps are fixations, the average cosine similarity of timesteps within the same event class is 0.92, while the average cosine similarity of timesteps across classes is 0.19. At the same time, we clearly struggle to separate the saccade class from the other classes. This indicates that

saccades are of a somewhat ambigous nature, which already has been stated in [1].

We conclude, that the DETRtime backbone is doing a good job at extracting meaningful features, that already separate the timesteps of different classes in the Euclidean space, which can then be handled by the transformer nicely. We ran experiments trying to predict the class of a time step based on the pretrained embeddings, showing very poor results compared to DETRtime.

In the course of this project we did not run any experiments of the DETRtime transformer on the artificially encoded timesteps with cosine-similarity as training objective. This is an interesting direction left for further research.

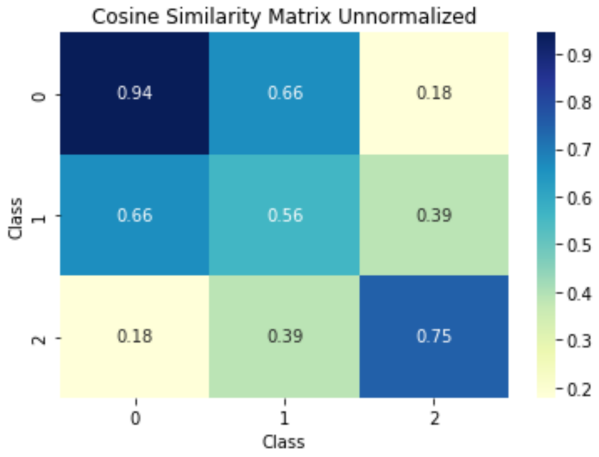

Figure 3.3: Cosine similarities of embeddings when being trained with a cosine-similarity objective and and Multilayer Perceptron to map the inputs.

3.2 Using DETRtime for Changepoint Detection

DETRtime has shown promising results in detecting eye movement events by introducing a new architecture and loss function in the regime of time-series segmentation [1]. However, DETRtime has certain limitations and can benefit from further improvements.

In this study, we propose modifications to DETRtime as proposed in [1] to address some of its limitations. Specifically, we focus on improving the detection of eye movement events by introducing a changepoint detection method and a postprocessing step. We evaluate the performance of our modified model on the dots-paradigm EEG dataset and compare it with the performance of the original DETRtime model. The metric used for the analysis is the macro F1-score, describing the classification performance of the three ocular events: fixation,

saccade, and blink.

3.2.1 Limitations of Event Detection in DETRtime

Despite achieving high macro F1 scores, DETRtime exhibits certain limitations resulting from its event detection approach for prediction. DETRtime segments time series data into labeled boxes by predicting start and end points for each event. However, the approach may lead to some limitations that can affect the accuracy and generalizability of the model.

One of the primary limitations of DETRtime is that the boxes generated by the model may overlap or have gaps where multiple classes or no class is predicted. This is problematic because each timestamp must belong to exactly one class. The current approach leads to ambiguity in the labeling of these timestamps.

To address this issue, the authors of DETRtime propose a heuristic approach where non-assigned classes are labeled as "fixation," and non-maximum suppression is used for timestamps that have been assigned to multiple classes. The authors found that this approach worked well when segmenting ocular events in EEG data because "fixation" was the dominant class. However, it is unclear whether this approach generalizes to other domains with different class distributions.

3.2.2 Introduction of Changepoint Detection

To overcome this limitation, we built upon DETRtime's strengths in attention mechanism and introduced a more natural prediction scheme for time-series segmentation. Specifically, we predict changepoints instead of bounding boxes, thereby offering a more straightforward approach to time-series segmentation. This approach is based on detecting changes in the time series, where each changepoint indicates a transition from one segment of the time series to another.

Each prediction consists of two entities: the first is a position p_i, which is a relative x-coordinate in the value range [0,1]. The second entity is the cross-entropy vector, indicating the probabilities for each class, including the non-object class. The class prediction indicates the class assignment of the timestamps after the changepoint.

Similar to DETRtime, we use the Hungarian loss as our objective function, computing a minimal matching of our predictions and targets. This objective can be further motivated by the theory of optimal transport as this loss minimized the monge objective [42]. For the class predictions, we use the cross-entropy loss. For the position prediction, we use the L1 loss.

Our approach offers several advantages over the event detection approach used in DETRtime. First, it is more natural because it directly detects changes

in the time series, which are the defining characteristics of many real-world phenomena. Second, it avoids the ambiguity of the event detection approach, where timestamps may be assigned to zero or multiple classes. Hence, it offers a more straightforward approach to time series segmentation, which can improve the generalizability of the model.

3.2.3 Prediction and Sequence Generation

Once all predictions have been made for a given sample using our changepoint detection approach, we generate the sequence as follows. First, we consider all changepoints for which the confidence of non-class objects is not the highest. This means that there is a class that has the highest confidence among all other classes.

Next, we sort the changepoints according to their position from left to right and map them onto the full length of the sample, which is 500 timestamps in our case. We then assign each timestamp coming after a changepoint to the class predicted for that changepoint until another changepoint appears. As our model has learned that each sequence always begins with a changepoint, we do not have the problem of unassigned changepoints at the beginning of a sequence.

Our approach also avoids the initial problems of DETRtime where changepoints can be non-assigned or assigned to multiple classes. By focusing on the highest-confidence class for each changepoint, we ensure that each timestamp is assigned to a single class and there is no ambiguity in the prediction.

3.2.4 Training and Hyperparameters

The model is trained for 250 epochs using a batch size of 32 and a learning rate of 1e-4. In order to balance the gradient information, we multiplied the L1 loss on the position by 20, as we observed that the magnitudes of the losses differed by a factor of 20. The full list of hyperparameters used in the training process can be found in the appendix.

During the training process, we observed a high variance in the metric score, even in the later epochs. This variance can be attributed to the model's instability towards false positive and false negative changepoints. This motivates the use of a postprocessing step.

3.2.5 Postprocessing

In this subsection, we describe a postprocessing step to address the instability towards false positives in the changepoint approach. False positives or false negatives in changepoint detection can result in misclassification of many timestamps

and can drastically reduce precision, especially for short classes such as "saccade" or "blink." To address this, we propose computing the posterior distribution of class lengths to identify class events with unlikely lengths.

To augment the definition of confidence for a changepoint, we multiply the predicted class confidence by the probability of the class lengths given the posterior length distribution of each class. This gives each changepoint prediction a new confidence measure. Using this measure, we filter out unlikely changepoints with unrealistic class lengths, such as a very long saccade.

To implement this postprocessing step, we iterate backward through the sequence of changepoints and compute the distance to the next changepoint, which is the length of the event. We then look up the probability from the posterior distribution of each class and compute the new confidence of this changepoint. If this confidence falls below a certain threshold, which is a newly introduced hyperparameter, then we evict this changepoint from the sequence.

By filtering out false positives through this process, we were able to increase precision, which is critical for short classes. We provide an example case where a false positive of the saccade class is predicted, successfully identified, and evicted by their postprocessing step, as shown in Image 3.4.

3.2.6 Results

The presented results demonstrate the impact of the proposed postprocessing step on the performance of the changepoint detection model. The macro F1 score increases from 0.83 to 0.87 with the inclusion of the postprocessing step. The precision of the saccade class improves from 0.51 to 0.80, and the recall of the fixation class increases from 0.96 to 0.99 with the postprocessing step. However, the recall of blinks and saccades slightly decreases. Overall, we can see an improvement of 0.04 by using the postprocessing procedure.

Although the DETRtime changepoint detection with postprocessing achieves a competitive score of 0.87, the original DETRtime as proposed in the paper outperforms the modified model by 0.05 margin, with a score of 0.92. The proposed changepoint detection method resolved the issue of ambiguity in the original DETRtime and improved its generalizability, but it also introduced new challenges, including unstable false negative and false positive predictions. Consequently, the results worsened, and the proposed modification did not result in an improvement over the original DETRtime. We thus conclude that the assumed independence in the box objective leads to better performance overall.

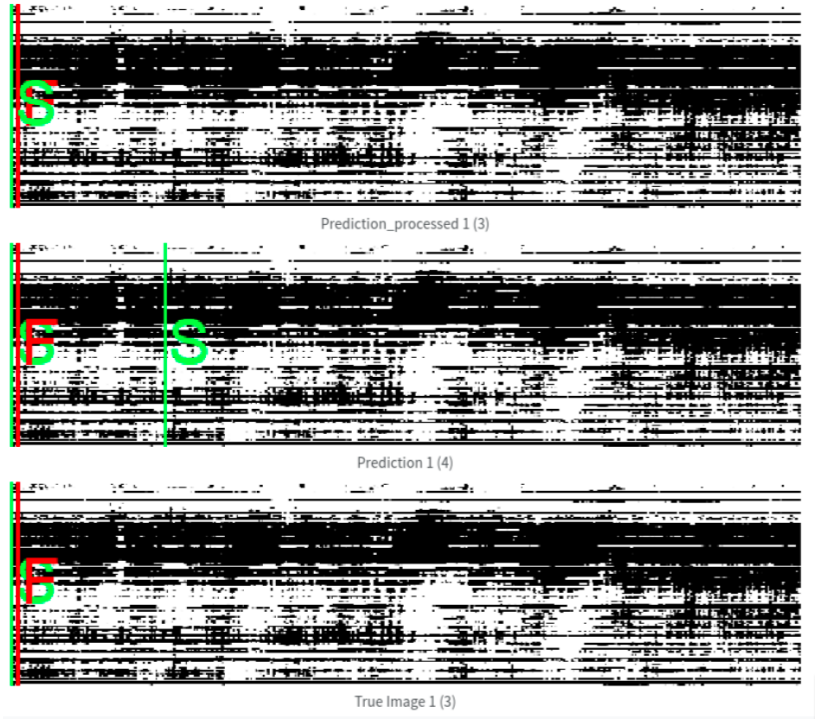

Figure 3.4: The image presents a comparison of the ground truth, prediction, and post-processed prediction. The bottom part of the image depicts the ground truth, while the middle part shows the initial prediction made by the model. It is evident that the prediction includes a false positive saccade changepoint, which misclassifies 80% of the timestamps in the sample. However, the top part of the image demonstrates the post-processed prediction, which successfully eliminates the false positive prediction and significantly improves the overall accuracy of the prediction

3.3 Improving Generalization through Self-Supervised Pretraining

We additionally investigate the topic of self-supervision in the EEG domain, inspired by successes in other domains such as computer vision and natural language processing. As is the case with DETRtime, initial approaches leveraged large labeled datasets to achieve initial successes. In any case, this approach is largely limited by the amount of available labeled data.
Recently, self-supervised learning has gained a foothold in specifically natural language processing (NLP) and computer vision as a pre-training objective in order to learn suitable features for an array of downstream tasks. Given the similarities in architectures used in these domains and DETRtime, we found it prudent to investigate a similar usage of common pre-training objectives.

3.3.1 Pre-training objective

Masked modeling and similar auto-regressive objectives have been leveraged as highly successful methods in NLP pretraining ([43], [44]). A portion of the input sequence is masked and held out and models are trained to predict the missing content. In NLP, these methods have performed well and there is ample evidence demonstrating good generalization of learned representations to downstream tasks.

The approach is generalized to more continuous settings such as vision by using auto-encoding techniques, where an encoder maps the input space to a latent space and a decoder reconstructs the input. With the rise of architecturally similar transformer architectures for the vision domain ([45]), combining this approach with the masked objective of NLP became more feasible.

So called masked auto-encoders, where additionally the input contains masked patches, where found to be similarly well-performing for transfer learning on downstream tasks ([46]). Still, the largely heavier spatial redundancy in the continuous vision setting represents an additional challenge for learning useful features, which can be overcome with masking large percentages of the input space.

Since the time series setting is similarly continuous with high local redundancies, we adapt a similar objective for the time series setting, by masking sections of the input signal and using various distance measures such as L1 distance or mean squared error (MSE) as proxies for reconstruction quality.

3.3.2 The architecture

We adapt the ViT transformer ([45]) for the time series setting to yield a masked autoencoder for the time series setting. The encoder works on unmasked portions of the input sequences to produce latent embeddings, which are used together with mask tokens by the decoder to reconstruct the vision output. Similar to the ([1]), we use normalized input samples consisting of 500 time steps for the training.

Reconstruction Objective The input is reconstructed by predicting the per-time step values of the input signals. The loss function then computes the mean squared error between the reconstructed and original values of the masked portions of the input signal.

Masking ratio We found a masking ratio of around 40% due yield the best results, providing better trade-off for the generally smaller redundancy of time series data.

3.3.3 Results and Discussions

We report on our best performing models an average L1 distance of around 187.5 per sample. At the chosen masking ratio, this corresponds to an average of 90 % the expected standard deviation on our normalized signal, which we found to be particularly unsatisfactory given the additional challenges of our reconstruction proxies on time-series data.

Given the expected high correlation between neighboring samples in EEG, one can argue that similarly other time-series based domains an interpolation is learned. Counter-acting this requires masking larger contiguous patches, but even for smaller size masks of around 20 timesteps (corresponding to 20ms), we find that this in turn negatively affects reconstruction. Since MSE or L1 loss already erroneously assume independent errors and fail to capture certain degrees of errors in time series ([47], [30]), we find the general distance-based reconstruction errors to be unsuitable for good representation learning on EEG.

Chapter 4

Improving Representations of DETRtime

4.1 Evaluation of BENDR on EEGEyeNet

The BENDR method has shown promising results in various classification tasks. However, it is essential to validate these results and extend them to non-classification tasks. This would provide a more comprehensive understanding of the algorithm's performance and potential applications.

EEGEyeNet contains EEG data with ocular events, as well as tasks of increasing difficulty, including classification and regression problems of distances and angles. It has been widely used as a benchmark for various EEG data analysis methods, making it an excellent choice for evaluating the performance of the BENDR method. This provides a comprehensive understanding of the algorithm's strengths and weaknesses and help identify areas for improvement.

4.1.1 BENDR Pretraining

To perform the pretraining, we randomly sample the input streams from the EEG data as described in Section 2. However, we perform less downsampling in the convolutional encoder in order to obtain a higher effective sampling rate than BENDR. This is done with the intention of testing downstream tasks that need higher temporal resolution.

We conduct pretraining experiments over multiple sequence lengths, and we also experiment with mask rates and span sizes. Additionally, we experiment with various numbers of encoder features, which represent the number of feature maps of the convolutional operations.

As shown in Table 4.2, we achieve similar loss values, which confirms the statement in the BENDR paper that the architecture is sequence length agnostic. After pretraining, we fine-tune on downstream tasks using multiple versions of BENDR. Surprisingly, we find that there is not much change in downstream task

performance among the different BENDR versions. Therefore, in the following section, we focus only on finetuning the BENDR version that achieved the lowest test loss.

BENDR Pretraining on EEGEyeNet				
Sequence Length	Mask Rate	Mask Span	Encoder Features	Test Loss
10000/84	0.1	5	512	3.78
4000/84	0.1	5	512	3.63
4000/84	0.065	5	512	3.82
4000/84	0.1	5	128	3.53
2000/42	0.1	5	256	3.33
2000/42	0.1	5	128	**3.21**

Table 4.1: Pretraining results of the BENDR architecture on all paradigms of the EEGEyeNet dataset. The sequence length denotes input sequence length/encoder sequence length

4.1.2 BENDR Finetuning

To evaluate the performance of the BENDR architecture on downstream tasks, we fine-tune the model by training a small MLP for each task. We use the BENDR convolutional encoder for inference to produce the "BENDR" embeddings, and freeze its weights without retraining them on the downstream task. We apply different loss functions depending on the specific task. For each downstream task, the same MLP architecture is used, consisting of 2 hidden layers with a dimension of 32 and an output layer with either 1 or 2 output logits, depending on the task. The MLP has a total of 46000 parameters. We choose the encoder with the smallest number of features tested (128) to reduce the complexity in the MLPs for the downstream tasks.

The LR task is considered the easiest and acts as a sanity check, as it is a problem that is already solved. The fine-tuned model achieves a comparable accuracy to the EEGeyenet, reaching an accuracy of 96.3 after training. The training converges quickly, with an accuracy of 96 achieved after only one epoch.

In addition, the utilization of BENDR representations in regression tasks has yielded promising results with respect to its generalization capabilities. Despite performing comparatively worse than EEGEyeNet models in the angle and amplitude tasks, the BENDR model outperforms the naive baseline. Furthermore, convergence in these tasks was observed after only a few epochs of training.

Lastly, the position task can be considered the most challenging task. Our fine-tuned models are slightly worse than the EEGeyenet model but have comparable performance.

The different fine-tuning tasks exhibit similar training behavior, with performance being comparable across all tasks with the EEGeyenet, although slightly worse. The fast training and convergence can be attributed to the small MLP and good representations by BENDR. Overall, the fine-tuning results demonstrate the effectiveness of the BENDR architecture for downstream tasks, with its good performance and fast training and convergence.

EEGEyeNet Tasks				
Model	LR	Angle	Amplitude	Position
BENDR	96.3	0.55	49.0	78.5
EEGEyeNet	98.8	0.33	30.7	70.2
Naive Baseline	52.3	1.90	74.7	123.3

Table 4.2: Here, the BENDR results refer to the pre-trained BENDR model, where after the frozen encoder we only fine-tune a simple MLP on the respective task. The EEGEyeNet result refers to the best model reported in the benchmark [3]. LR performance is given in Accuracy, Angle in radians, and Amplitude and Position in RMSE.

4.2 DETRtime for Region Learning

In preliminary experiments we found DETRtime in its original region based formulation to be the best performing segmentation model. With regards to representation learning a contrastive predictive coding [48] based approach showed promising results on various downstream tasks.

In the following we propose a unified approach towards using segmentation abilities of DETRtime to guide representation learning.

4.2.1 Transferring BENDR Ideas

The high level ideas surrounding the representation learning in BENDR are based on contrastive predictive coding (CPC) [48]. In these CPC based models, an initial encoder E is used to map observations x_t to a sequence of latent representations $z_t = E(x_t)$. Temporal resolution of these latent representations may differ here, as is the case in [30]. An autoregressive model M then uses the latent encodings to construct a context representation $c_t = M((z_{\leq t})$. In BENDR, the corresponding $z_{\leq t}$ is replaced with whole sequence of latent encodings $(z_t)_t$.

Instead of direct future predictions, the goal is then to model the density corresponding to the mutual information between x_{t+k} and c_t. x_{t+k} may variously

be replaced with z_{t+k} s.t. the

$$f_k(x_{t+k}, c_t) \propto \frac{p(x_{t+k}|c_t)}{p(x_{t+k})} \tag{4.1}$$

Any positive scoring function may be used in these cases. Whereas the original paper proposes a bilinear function [48], BENDR [30] [29] has used cosine similarity as a scoring function.

Training Objective Direct evaluation of $p(x)$ and $p(x|c)$ is not tractable, however, one may use samples from these distributions for noise-contrastive estimation [49] to jointly optimize the encoder and autoregressive model. Given a set of samples X consisting of a single positive sample $p(x_{t+k}|c_t)$ and negative samples from the proposal distribution $p(x_t)$, optimizing the following objective

$$\mathcal{L}_N = \mathbb{E}_X \left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)} \right] \tag{4.2}$$

will result in a correct estimation of the density ratio in 4.1.

We combine these ideas with the general ideas underlying change point detection and segmentation[50] [51], whereby there exists an underlying (latent) process that suddenly changes at certain times. Under this viewpoint, a signal may be discretized into variably sized segments.

Region Learning We note that such segments may generally be seen as context representations $c_{I_{t'}}$ where $I_{t'}$ corresponds to an interval or region possibly containing multiple time steps. In particular, the general hidden state outputs of DETRtime may represent such variable-sized regions. Concretely, we reformulate the context representations as $(c_{I_{t'}}, I_{t'}) = M((z_t))_i$, where the model M predicts an interval $I_{t'}$ and its corresponding context region $c_{I_{t'}}$. Under this formulation, we assume the generating distribution for a time step t to be $p(x_t|c_{I_{t'}})$ where $t \in I_{t'}$. Alternatively, one can directly model the ratios for $p(x_t|c_{I_{t'}}$ through a different sampling procedure.

For our model we implement the first approach by introducing an additional encoding E for $t \in I'_t$ s.t. $c_t = E_t(c_{I'_t})$ following an assignment of t to regions $I_{t'}$ based on predicted confidence to augment the DETRtime objective with BENDR's contrastive loss.

4.2.2 The Region Learning Model

Our proposed model relies on learned accurate segmentations. In lieu of suitable fully self-supervised segmentation approaches, we thus first rely on supervised approaches to achieve good segmentation results.

The original DETRtime loss is used to train a transformer to predict discrete segments in a time-series signal x_t. The corresponding region representations and boundaries are then used to recover the latent encodings c_t at each time-step for the BENDR objective by simply assigning the latent embedding of a region to corresponding time-steps and using a simple MLP to recover the context encoding c_t.

One can variously train the supervised and unsupervised objectives jointly or separately, however, the model does rely on initial supervised training to yield meaningful segmentations. It is possible to only rely on partially labelled datasets in training.

The intent is that labelled segments will guide the learning of meaningful latent context representations in this way for further downstream tasks. We argue that this is especially the case for cases where downstream tasks can be meaningfully related to a classic segmentation task, as is the case in EEG-based eye-movement segmentation where saccades correspond to the only regions where eye movements should be detected.

Chapter 5

Evaluation and Outlook

In this work, we explored and evaluated the challenges of time series tasks with the DETRtime architecture and proposed a pathway towards a semi-supervised region learning model for (EEG) data by incorporating ideas from BENDR and DETRtime.

We evaluated potential reformulations of DETRtime's detection objective by evaluating a change towards change point detection. Despite possibly benefits over the initial independent region objective, we found the original formulation to be more robust and wellperforming.

The DETRtime backbone embeddings were evaluated using different similarity measures to better understand what is needed for effective time series segmentation. We found that DETRtime is able to effectively separate time steps from separate classes, and pretraining with a cosine similarity objective did not improve performance. In fact, we found the additional cosine similarity objective to have an adverse effect on performance and observed that DETRtime achieves better class separation specifically for the downstream tasks more important saccade class.

We also explored different representation learning approaches for time series data. We found that unlike in e.g. NLP domains where simple reconstruction objectives suffice, more complicated techniques such contrastive predictive coding perform better on EEG data due to the specific challenges of high local redundancies and sparse long-term dependencies that time series data presents.

We note that DETRtime's initial segmentation can be augmented with CPC to provide a semi-supervised representation learning based on region representations. Currently, much of the evaluation of our model is outstanding, including the exploration of different training modalities to achieve the the intended semi-supervised learning.

We have prepared a host of downstream tasks for evaluation and further work will focus on in-depth evaluation of these tasks. Our proposed semi-supervised region learning model has the potential to improve downstream tasks performance, e.g. the accuracy of time series segmentation in EEG data, and we hope our work

will inspire further exploration of this approach in the field.

Bibliography

[1] L. Wolf, A. Kastrati, M. B. Płomecka, J.-M. Li, D. Klebe, A. Veicht, R. Wattenhofer, and N. Langer, "A deep learning approach for the segmentation of electroencephalography data in eye tracking applications," 2022. [Online]. Available: https://arxiv.org/abs/2206.08672

[2] C. R. Michael Cohen, Christian E. Elger, "Reward expectation modulates feedback-related negativity and eeg spectra," *Neuroimage, 35(2):968-978*, 2007.

[3] A. Kastrati, M. B. Płomecka, D. Pascual, L. Wolf, V. Gillioz, R. Wattenhofer, and N. Langer, "Eegeyenet: a simultaneous electroencephalography and eye-tracking dataset and benchmark for eye movement prediction," 2021. [Online]. Available: https://arxiv.org/abs/2111.05100

[4] "Hans berger (mediziner) – wikipedia," https://de.wikipedia.org/wiki/Hans_Berger_(Mediziner), (Accessed on 02/21/2023).

[5] S. Tong and N. V. Thankor, *Quantitative EEG analysis methods and clinical applications*. Artech House, 2009.

[6] A. A. Chaaraoui, P. Climent-Pérez, and F. Flórez-Revuelta, "A review on vision techniques applied to human behaviour analysis for ambient-assisted living," *Expert Systems with Applications*, vol. 39, no. 12, pp. 10 873–10 888, 2012.

[7] G. Okada, K. Masui, and N. Tsumura, "Advertisement effectiveness estimation based on crowdsourced multimodal affective responses," in *Proceedings of the IEEE Conference on computer vision and pattern recognition workshops*, 2018, pp. 1263–1271.

[8] P. Majaranta and A. Bulling, "Eye tracking and eye-based human–computer interaction," in *Advances in physiological computing*. Springer, 2014, pp. 39–65.

[9] M. K. Eckstein, B. Guerra-Carrillo, A. T. M. Singley, and S. A. Bunge, "Beyond eye gaze: What else can eyetracking reveal about cognition and cognitive development?" *Developmental cognitive neuroscience*, vol. 25, pp. 69–91, 2017.

[10] A. Bulling, J. A. Ward, H. Gellersen, and G. Tröster, "Eye movement analysis for activity recognition using electrooculography," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 4, pp. 741–753, 2010.

[11] R. Nakashima, Y. Fang, Y. Hatori, A. Hiratani, K. Matsumiya, I. Kuriki, and S. Shioiri, "Saliency-based gaze prediction based on head direction," *Vision research*, vol. 117, pp. 59–66, 2015.

[12] C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry," in *Matters of intelligence*. Springer, 1987, pp. 115–141.

[13] K. Krafka, A. Khosla, P. Kellnhofer, H. Kannan, S. Bhandarkar, W. Matusik, and A. Torralba, "Eye tracking for everyone," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2176–2184.

[14] J. Son, L. Ai, R. Lim, T. Xu, S. Colcombe, A. R. Franco, J. Cloud, S. LaConte, J. Lisinski, A. Klein *et al.*, "Evaluating fmri-based estimation of eye gaze during naturalistic viewing," *Cerebral Cortex*, vol. 30, no. 3, pp. 1171–1184, 2020.

[15] S. LaConte, S. Peltier, K. Heberlein, and X. Hu, "Predictive eye estimation regression (peer) for simultaneous eye tracking and fmri," in *Proc. Intl. Soc. Magn. Reson. Med*, vol. 2808, 2006, p. 13.

[16] T. P. O'Connell and M. M. Chun, "Predicting eye movement patterns from fmri responses to natural scenes," *Nature communications*, vol. 9, no. 1, pp. 1–15, 2018.

[17] A. Kastrati, M. B. Plomecka, R. Wattenhofer, and N. Langer, "Using deep learning to classify saccade direction from brain activity," in *ACM Symposium on Eye Tracking Research and Applications*, ser. ETRA '21 Short Papers. Association for Computing Machinery, 2021.

[18] M. L. Mele and S. Federici, "Gaze and eye-tracking solutions for psychological research," *Cognitive processing*, vol. 13, no. 1, pp. 261–265, 2012.

[19] A. Bulling and D. Roggen, "Recognition of visual memory recall processes using eye movement analysis," in *Proceedings of the 13th international conference on Ubiquitous computing*, 2011, pp. 455–464.

[20] R. J. Jacob and K. S. Karn, "Eye tracking in human-computer interaction and usability research: Ready to deliver the promises," in *The mind's eye*. Elsevier, 2003, pp. 573–605.

[21] E. M. Reingold, "Eye tracking research and technology: Towards objective measurement of data quality," *Visual cognition*, vol. 22, no. 3-4, pp. 635–652, 2014.

[22] R. Zemblys, D. C. Niehorster, and K. Holmqvist, "gazenet: End-to-end eye-movement event detection with deep neural networks," *Behavior research methods*, 2018.

[23] R. Barea, L. Boquete, M. Mazo, and E. López, "System for assisted mobility using eye movements based on electrooculography," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 10, no. 4, pp. 209–218, 2002.

[24] F. Behrens, M. MacKeben, and W. Schröder-Preikschat, "An improved algorithm for automatic detection of saccades in eye movement data and for calculating saccade parameters," *Behavior research methods*, vol. 42, no. 3, pp. 701–708, 2010.

[25] A. Bulling, J. A. Ward, H. Gellersen, and G. Tröster, "Eye movement analysis for activity recognition," in *Proceedings of the 11th international conference on Ubiquitous computing*, 2009, pp. 41–50.

[26] K. Pettersson, S. Jagadeesan, K. Lukander, A. Henelius, E. Hæggström, and K. Müller, "Algorithm for automatic analysis of electro-oculographic data," *Biomedical engineering online*, vol. 12, no. 1, pp. 1–18, 2013.

[27] K. Kleifges, N. Bigdely-Shamlo, S. E. Kerick, and K. A. Robbins, "Blinker: Automated extraction of ocular indices from eeg enabling large-scale analysis," *Frontiers in neuroscience*, vol. 11, p. 12, 2017.

[28] C. Fefferman, S. Mitter, and H. Narayanan, "Testing the manifold hypothesis," 2013. [Online]. Available: https://arxiv.org/abs/1310.0425

[29] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *CoRR*, vol. abs/2006.11477, 2020. [Online]. Available: https://arxiv.org/abs/2006.11477

[30] 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," *CoRR*, vol. abs/2101.12037, 2021. [Online]. Available: https://arxiv.org/abs/2101.12037

[31] D. Guijo-Rubio, A. M. Durán-Rosal, P. A. Gutiérrez, A. Troncoso, and C. Hervás-Martínez, "Time-series clustering based on the characterization of segment typologies," *IEEE Transactions on Cybernetics*, vol. 51, no. 11, pp. 5409–5422, 2021.

[32] R. Ma, A. Ahmadzadeh, S. F. Boubrahimi, and R. A. Angryk, "Segmentation of time series in improving dynamic time warping," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 3756–3761.

[33] D. Trabelsi, S. Mohammed, F. Chamroukhi, L. Oukhellou, and Y. Amirat, "An unsupervised approach for automatic activity recognition based on hidden markov model regression," *IEEE Transactions on Automation Science and Engineering*, vol. 10, no. 3, pp. 829–835, 2013.

[34] A. Chinea, "Understanding the principles of recursive neural networks: A generative approach to tackle model complexity," *CoRR*, vol. abs/0911.3298, 2009. [Online]. Available: http://arxiv.org/abs/0911.3298

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[36] 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.

[37] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020. [Online]. Available: https://arxiv.org/abs/2002.05709

[38] N. L. Andreas Pedroni, Amirreza Bahreini, "Automagic: Standardized preprocessing of big eeg data." *Neuroimage, 200:460-473*, 2019.

[39] K. K.-D. Luca Pion-Tonachini and S. Makeig, "Iclabel: An automated electroencephalographic independent component classifier, dataset, and website." *NeuroImage, 198:181–197*, 2019.

[40] N. Hollenstein, M. Troendle, C. Zhang, and N. Langer, "Zuco 2.0: A dataset of physiological recordings during natural reading and annotation," *CoRR*, vol. abs/1912.00903, 2019. [Online]. Available: http://arxiv.org/abs/1912.00903

[41] K. P. Miika Toivanen and K. Lukander, "A probabilistic real-time algorithm for detecting blinks, saccades, and fixations from eog data." *Journal of Eye Movement Research, 8(2)*, 2015.

[42] D. Feyel and A. S. Ustunel, "The monge-kantorovitch problem and monge-ampere equation on wiener space," 2003. [Online]. Available: https://arxiv.org/abs/math/0306323

[43] A. Radford and K. Narasimhan, "Improving language understanding by generative pre-training," 2018.

[44] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[45] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *CoRR*, vol. abs/2010.11929, 2020. [Online]. Available: https://arxiv.org/abs/2010.11929

[46] K. He, X. Chen, S. Xie, Y. Li, P. Doll'ar, and R. B. Girshick, "Masked autoencoders are scalable vision learners," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15 979–15 988, 2021.

[47] F. Rivest and R. Kohar, "A new timing error cost function for binary time series prediction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 1, pp. 174–185, 2020.

[48] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018. [Online]. Available: http://arxiv.org/abs/1807.03748

[49] Z. Ma and M. Collins, "Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency," 2018. [Online]. Available: https://arxiv.org/abs/1809.01812

[50] E. Caldarelli, P. Wenk, S. Bauer, and A. Krause, "Adaptive Gaussian process change point detection," in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 2542–2571. [Online]. Available: https://proceedings.mlr.press/v162/caldarelli22a.html

[51] E. J. Keogh, S. Chu, D. M. Hart, and M. J. Pazzani, "Segmenting time series: A survey and novel approach," 2002.