Zürcher Hochschule
für Angewandte Wissenschaften

School of Engineering

VT1 | MSE Data Science

Evaluation of Data Augmentation Strategies for Motor Imagery BCI Classification Tasks Based on EEG Data

Author Manuel Weiss

Advisor Ricardo Chavarriaga

Date 31.01.2023

Abstract

The field of machine learning based electroencephalogram (EEG) classification has received increasing attention in research in recent years. However, the limited amount of available training data especially restricts the use of deep learning based models in real-world applications of brain-computer interface (BCI) systems, which allow humans to control or communicate with devices only by their brain activity. One method of addressing data scarcity, known as data augmentation, is to increase the amount of available data and its diversity by applying realistic transformations to the original data set to obtain new samples. The impact of transformation-based augmentation techniques in different experimental settings, however, is often still unknown. Therefore, this project compares three different augmentation methods, namely Channels Dropout (CD), FT Surrogate (FTS) and Smooth Time Mask (STM), applied to two Motor Imagery (MI) data sets in experiments based on single and multi-person data. The relative change in classification accuracy compared to self-conducted baseline experiments was evaluated with a shallow and a deep Convolutional Neural Network (CNN) as well as with a Support Vector Machine (SVM). For the different combinations of experimental scenarios, the methods show diverging levels of classification performance. Achieved accuracy gains of 8.56% with STM and 7.6% with FTS contrast with a decline in accuracy of the same methods in slightly different scenarios such as changing the depth of the CNN. Nevertheless, tendencies to reduce overfitting or to narrow differences in performance between subjects could be observed. Eventually, based on the results obtained, improvements to the current experiments such as additional preprocessing to further reduce the variance between subjects are discussed and prioritised for future work.

Contents

Chapter 1

Introduction

1.1 Problem Statements

The analysis of brain activity encounters numerous applications within and outside the clinical domain [1]. A very popular method of recording brain dynamics is to measure the electrical activity of the brain using electrodes placed on the scalp, which is called electroencephalography (EEG) [2]. The recorded EEG signals can be used for a variety of tasks such as detecting epilepsy, monitoring sleep phases or for brain-computer interfaces (BCIs) [2].

In this project, EEG signals are analysed solely for BCIs. According to Iturrate, Chavarriaga and Millán [3], BCIs can be seen as systems that translate brain activity patterns into commands that can be executed by artificial devices. Also Leeb [4] defines BCIs as the idea of controlling a machine by «thinking» rather than by manual operation.

A sub-area in which BCI systems can be applied are motor imagery (MI) tasks, which is the BCI paradigm all experiments of this work are based on. The MI term refers to the mental imagination of body movements [5;6]. The signal captured during this mental tasks can then be used in a supervised learning classification problem [7;8]. Models which reliably recognise body movements based on brain activity could in future enable people to operate prostheses, exoskeletons or wheelchairs, control games and typewriter applications [3]

Nowadays, BCI systems mainly rely on machine learning (ML) approaches [9;10]. But still, EEG signals are complex, non-stationary, high dimensional and they lack of a low signal to noise ratio [1;11;12].

The application of deep learning (DL) has led to further advances in the use of machine learning for MI tasks and surpassed the performance of classical ML models such as the Support Vector Machine (SVM) [13;7;8]. Especially with the use of Convolutional Neural Networks (CNN) and their ability to artificially generate valuable features, preprocessing is simplified and less expertise is required [14]. However, all the advantages of modern DL approaches come at the cost of a large amount of data needed for training, which poses a new limitation in classifying EEG signals [15]. Recoding EEG data is time consuming, requires in-depth knowledge and determined volunteers need to be trained specifically for the MI task at hand [16;1]. All these circumstances lead to a general shortage of data in this area [1].

This project focuses on transformation-based data augmentation to tackle the problems of data scarcity. Augmentation techniques address the lack of data by artificially increasing the distribution of training data and ideally improve a model's generalisation capabilities across multiple data sets [17;18]. They should also reduce the problem of exactly modelled training data, known as overfitting [17;18]. The literature proposes and assesses many augmentation techniques for EEG data [19;20;21;22;23;24;25;26]. But only in a few papers, augmentation techniques are compared in a broader scale [27]. Most of the literature focuses on the theoretical introduction of new augmentation strategies rather than on investigating their effects in different experimental contexts. The goals of this project is therefore to conduct extensive experiments with the three different augmentation techniques Channels Dropout (CD), FT Surrogate

(FTS) and Smooth Time Mask (STM) to evaluate the transferability of this methods to different tasks. The techniques will be assessed on two MI data sets, with two CNN architectures of contrasting depth and a SVM, optimized for a single person or for all people in a data set.

1.2 Related Work

Although the use of augmentation techniques for EEG classification tasks is still in its infancy, there are several papers in the literature on their application for motor imagery BCIs[27;28;29;30;31;32;33]. This emphasises the interest and attention this area currently experiences in the data and neuroscience community. In addition, there are several papers proposing new augmentation methods for EEG data in general, which can potentially be transferred to motor imagery classification tasks. Schwabedal et al.[19], Mohsenvand[23] and Cheng et al.[24] propose FT Surrogate and Smooth Time Mask initially for sleep stage detection and then for motor imagery tasks. Saeed et al.[20] and Deiss[22] introduce spatial techniques like Channels Shuffle or Channel Symmetry for the detection of diseases such as seizures, while Krell and Kim[25] use sensor rotation to classify events in brain activity. Wang et al. suggest a Gaussian noise based method for emotion recognition[21]. Finally, Alwasiti and Yusoff[34] use a procedure called Mixup Augmentation for a motor imagery classification task, primarily introduced by Zhang et al.[26] for image classification.

Nevertheless, no general statement can currently be made about the effectiveness of augmentation techniques for EEG data, and as Rommel et al.[27] emphasises, extensive experiments are still necessary. Also, all existing investigations underlay on a fairly different setting with its distinct focus. As an example, whereas Freer and Yang[35] concentrate on the effects of augmentation techniques on a specific data set, George et al.[31] focus on the evaluation of augmentation methods with an newly introduced model architecture and Zhang et al.[26] consider the impacts of a distinct technique.

Recently, Rommel et al.[27] conducted an extensive comparison between different augmentation techniques such as Sign Flip, Gaussian Noise, Sensor Rotation or Channels Dropout. In the paper, the shallow CNN of Schirrmeister et al.[36] was applied on the four-class MI data set of the *BCI IV 2a* competition[37] as well as to a sleep stage detection task on the *Sleep PhysioNet* data set[38]. As the model and data set are publicly available, a reconstruction of the experimental results can be carried out for comparison. It also allows to extend the investigations to different experimental settings such as changing the underlying data sets or models to gain a deeper understanding.

What is still unknown, is whether the results of the presented papers can be transferred to other experimental scenarios of motor imagery classification tasks and how sensitive these results react to changes. Specifically, this means to different data sets, models or changing training scenarios, such as optimising a model for a single person or for a group of different people.

1.3 Goals

The main objective of this work is to perform extensive augmentation experiments with a selected set of augmentation techniques in changing experimental contexts in order to draw further conclusions about the effectiveness of the methods for an motor imagery classification task. The aims of the experiments can be segmented in three different goals:

1. Assessing the change in classification accuracy when applying a specific augmentation technique to different data sets in order to validate the transferability of a method to new motor imagery classification tasks.
2. Evaluating augmentation methods for different training scenarios which result in discrete optimisation problems and a changing level of difficulty for the model.
3. Comparing the applied techniques between neural based and classical machine learning approaches.

All this investigations and experiment should answer the question about an augmentation strategy's adaptation capabilities to different scenarios. Then if evidence can be found, that some methods are likely to increase the performance for motor imagery tasks, this could speed up the decision and implementation processes in EEG classification projects. Furthermore, a statement should be made about the extent to which the evaluated augmentation techniques improve the training process in EEG classification tasks. In other words, do they increase classification performance, make a model more robust to inter-subject variability, or do they improve the stability during training? Lastly, this project should provide advice on additional experiments needed to assess the effectiveness of augmentation techniques for EEG data based on the new results of this work.

Chapter 2

Theoretical Introduction

2.1 Electroencephalography

Electroencephalography (EEG) is a prevalent approach to extract information of brain dynamics [39]. While the signal was mainly used to identify neurological diseases, such as epilepsy, brain tumours, head injury and sleep disorders, it is nowadays considered as the most predominantly employed input signal in BCI systems [40;41;1].

The signal is recorded by placing electrodes on the scalp that measure the electrical activity of the brain over time, which is called an electroencephalogram [2]. More precisely, each electrode records electrical activities describing the amplitude measured in V or μV over time [42]. An EEG channel is then obtained using a differential amplifier, which takes two electrical inputs from two electrodes and outputs the difference between the two inputs [42]. In other words, an EEG channel describes the relative change in electrical activity between two referencing electrodes [43]. The two reference electrodes that form an EEG channel depend on the placement of the electrodes on the scalp, which is called montage [43]. Many different types of montages exist. The two types appearing in this work are the monopolar and the bipolar montage [43]. In a monopolar recording, an EEG channel describes the voltage difference between an electrode placed on the scalp and a single reference electrode placed on the earlobe called a ground electrode [43]. This results in as many EEG channels as electrodes placed on the scalp. In contrast, in a bipolar recording, a channel is formed by the difference between two adjacent electrodes on the scalp [43]. Depending on the strategy to select the referencing electrode, this often results in less channels than electrodes. Ultimately, taken all channels together, an EEG signal can also be described as two dimensional data formulated by its shape of $(c \times t)$ where c defines the number channels, and t expresses the time [39;36].

EEG signal can obtain a low signal to noise ratio depending on the application. The term noise refers to unwanted signals or frequencies within the combined input of all channels. Noise in the signal can have multiple reasons. Electrodes may detect different base activities in the brain that occur due to environmental perceptions or other vital processes, which results in similar amplitudes and may be undesirable for motor image classification [1;44;45]. Another reason can be different placements of the electrodes on the scalp [44;45].

EEG signals are non-stationary [11;12]. The term non-stationarity describes the characteristic that the statistical values of a signal such as mean, variance and covariance change over time [46]. This makes it challenging for model to classify data recorded over time (e.g. different days).

Finally, EEG signals not only comprise a high variability for data recorded from a single person, but also for data acquired from different people [47]. A person from which EEG data is recorded is called a subject [48]. According to Cheng et al. [24], data recorded from distinct subjects can be interpreted as different domains and subject-specific features are an integral part of EEG signals. The resulting

variance between subject is called inter-subject variability [24]. This phenomenon is referable to the physiological differences between humans [47].

2.2 Motor Imagery Data Set

In the literature, in neuroscience and in the field of machine learning, domain specific terms are sometimes used ambiguously [1]. Therefore this section aims to clarify the terminology of motor imagery data sets used in this report.

Motor imagery data is collected from subjects in one or more sessions [49;50]. A session is a finite collection of EEG data from a subject, which also includes the fixation and removal of electrodes [48]. A session is further divided into runs in which the subject has to solve one or more tasks, called trials, which are the atomic entities to which a label can be assigned [51].

EEG signals can be described as a continuous incoming stream of values at a given time point [52]. As soon as the defined trials are executed, markers are assigned to the signal indicating at which point in time a certain task was executed [1]. Then a label is assigned to every marker, according to the motor imagery task performed by the subject [1].

Since the execution of a task can take several seconds, it would not be representative to assign a label to a single point in time [52]. For this reason, a window of defined length is assigned to a marker, which serves as input for a classifier or for a preprocessing pipeline [1]. Such windows are called epochs in the neuroscience domain. The term epoch can thus lead to slight confusion with the term epoch from the machine learning terminology, in which it means a pass through the train data set when training a model [1]. Figure 2.1 shows the described possible hierarchical structure of a motor imagery EEG data set.

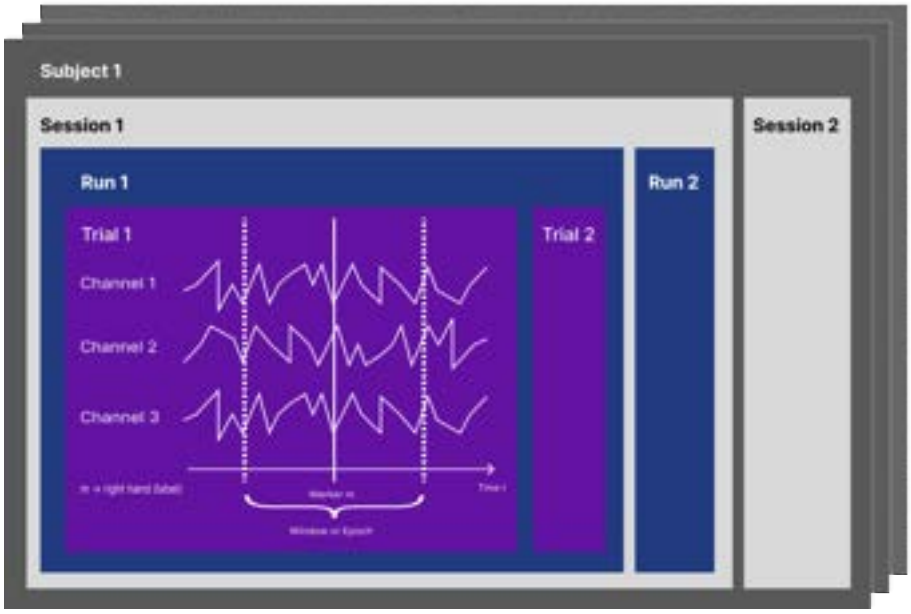


Figure 2.1: Motor Imagery Data Set Structure

2.3 EEG Preprocessing

Due to the characteristics of EEG signals described in section 2.1, preprocessing is often applied before classifying a data set. An often exploited family of technique to preprocess EEG are filters. The main idea of filtering EEG signal is to remove unnecessary signals or frequencies while trying to keep the relevant information [53;54]. High- and Low-Pass filter, Linear-Adaptive-Filter or Kalman-Filter are just some existing in the literature [55;54;56;53;57]. Noise in the signal can originate from different electrode placements, the used appliances, the experimental environment or from base activities in the brain of the subject itself [58;1;59;60]. However, filtering is often a complex task because noise and relevant information are often overlapping [55;54]. A successfully applied filter for EEG data is the band pass filter [61;62;58;36]. In the literature, frequencies below 4 Hz are often considered as noise for motor imagery tasks and could be eliminated by the filter [62;37;36]. Regarding Gatton [63], the signal in this low frequency regime likely emerge from eye movements by the subject. However, since the frequency ranges may refer to different actions performed by the subject, the frequency limits should be set according to the application and the classes to be predicted for a given data set.

EEG data is scarce in most publicly available data sets. Therefore, an efficient use of the data is vital to achieve reasonable results with data intensive model families such as neural networks. Data efficiency refers to the ability to learn in complex domains without a large amount of data, or to pull out more information of a given data set if training data is scarce [64;65]. One method to increase the number of samples, proposed by Schirrmeister et al. [36], is to cut the recorded trials into multiple crops. This method was inspired by similar procedures in object recognition within images [66;67]. Formally, each trial $X^j \in \mathbb{R}^{E \times T}$ with a number of E electrodes and T time steps is divided into crops of size T' time steps [36]. A resulting crop is defined as $C^j = \{X^j_{1..E,t..t+T'} | t \in 1..T-T'\}$ with $T-T'$ as the number of crops or new samples per trial [36]. Finally, to each evolved sample the same label y_i is assigned, as it was for the original trial [36]. As Schirrmeister et al. [36] points out, the ultimate consequence of this procedure is that the model is forced to use features that are present in all crops and cannot rely on the global temporal structure of an experiment. Another effect of cutting a trial into several smaller windows is the dimensionality reduction per sample. This can speed up training and simplifies the model's task to approximate the underlying distribution. However, this approach can also be disadvantageous when a large context of the data is required for discrimination, hence if the EEG data used is highly dependent on the time domain.

2.4 Training Scenarios

EEG data is usually recorded from multiple subjects, and due to the hierarchical structure, different training scenarios for a motor imagery classification task exist. For instance, a tuned model can be optimised for the body movement imagination of a single person or, with increasing complexity, for a group of people. In other words, the model can be trained on different distributions. Either on the distribution of a single subject or on a larger distribution of all subjects in a training data set. Ultimately, this relates to the question about the model's subject-invariance or how effectively it can be applied to new subjects, also known as generalisation over subjects [24]. The term generalisation in machine learning refers to the ability of a model to adapt to unseen data or data sets [68]. In this work, it is also used to describe the adaptation capabilities of a model across different scopes of a data set, such as subjects.

This work is concerned with whether an augmentation technique can advantageously expand the training distribution for a single subject only or whether it can make a model more subject-invariant. One possible way to test a model's generalization capabilities is to split the data set differently into train-, valid- and test set, allowed by the hierarchy of a MI data set described in section 2.2. For this project, the different splits applied are called split strategies. Possible split strategies are discussed in the upcoming paragraphs.

First of all, a data set can be divided by subjects. In the literature, a distinction between training a model on a single subject is called intra-subject classification, while inter-subject classification refers to

training on multiple subjects[69;70;71]. Intra- and inter-subject classification often concerns about the magnitude of variance motor imagery data contains. In an inter-subject scenario, the within-subject variability refers to question if a subject is capable to imagine body movements similarly such that the same amplitudes in the signal are present over time[51;69]. Evidently, the within-subject variance also depends on the amount of training a subject received before performing a certain MI task[51;69]. In contrast, in an inter-subject scenario, the between-subject variability refers to how similarly our brain emits a signal for a particular body movement and how similarly two subject learned a distinct task[51;69]. Because inter-subject classification can be used to evaluate a model's generalisation capabilities between subjects, it is essential to include a subject in the test set which the model has never seen before in order to assess the model performance appropriately[70;71;27]. Alternatively, the performance of the model in an intra-subject environment could be assessed by averaging over the results obtained by the individual subjects of a data set[69].

Further a data set can be split up by different sessions. This refers to the question of whether a trained model focuses on the correct information in the data rather than random background noise caused by the subject's current mental state, which can vary from day to day[50]. This scenario can be used, for example, to assess how well a model generalises for a subject between days. But also for this method, it is important that the final test set includes sessions which the model has not seen before[50;27]. For this strategy, one session is often used for training and a second for testing[50]. The division of a data set by session is called cross-session classification[48].

A data set can also be decomposed by runs[50]. This type of division can be used in a within-session classification scenario[48]. In this setting, runs can be divided into the different sub data sets for training, validating and testing[48]. If still multiple sessions were recorded, the performance of a model family can be evaluated by calculating the average over the performances obtained by the different sessions[48]. Within-session experiments contain the least amount of variance, which is why a higher performance of the model is expected. It helps to solve the question if ML algorithms are capable to classify EEG data. In the view of the author of this report, within-session classification is more applicable to experiments (e.g. variance of EEG data, signal to noise ratio or the effect of a subject's mental state) rather than practical use cases of EEG classification because of the unevaluated generalisation of the model over time.

Lastly, as in usual supervised learning classification problems, a data set could also be divided by samples[48]. This strategy can be chosen if more flexibility is required or none of the above general strategies is applicable for a given experiment. However, combining samples from multiple sessions to a test and validation set could likely result in double dipping and an overestimation of performance, which is why this strategy should be considered carefully and only if required by the application. Double dipping refers to train and evaluate a model on the same data set with resulting high statistical performance and possibly weak generalisation due to overfitting[72].

In summary, EEG data sets can be divided by...

- ...subjects with a distinction between intra- or inter-subject classification.
- ...sessions with a distinction between within- and cross-session classification.
- ...runs which can be used to divide the data set for a within-session classification
- ...samples which gives the most flexibility but is potentially leads to a double dipping problem.

Also, by choosing different strategies, arbitrary combination can be achieved depending on the requirements of the experiment or application. Therefore, it is important to specify which strategies are used for a set of experiments in order to differentiate between obtained performance results and in favour of reproducibility.

2.5 Metrics

In the interest of reproducibility and meaningfulness of the experiments, it is essential to name the metrics used and explain why they were chosen [1]. There are several opinions in the literature on how to select expressive metrics for a supervised learning classification task. On the one hand, a metric should accurately assess the real classification performance of a model. On the other hand, it should ensure the interpretability of the results for humans [68;73;74]. Ng [68] suggests to use a single metric for an efficient comparison between consecutive experiments, whereas Doshi-Velez and Kim [74] emphasise the inability of a single metric, such as the accuracy, to describe a real world problem [74]. In this report a combination of both approaches is applied. First, a single primary metric is identified for a straightforward, elementary comparison between experiments. Second, additional metrics should be computed to allow a more in-depth analysis of a change in classification performance, differences between classes of a data set or the sensitivity of a model to hyperparameter changes. The aim of this section is to present possible metrics that can be used in the experiments conducted for this project.

2.5.1 Accuracy

Accuracy is the most commonly used performance measure in supervised learning classification problems because of its simple interpretability and straightforward calculation [75;76]. However, it is often criticised for not being able to capture all facets of a model's classification performance and for failing with imbalanced data sets [77;78]. Accuracy is defined by [79;78]:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

For binary classification, accuracy can also be defined as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

2.5.2 Precision, Recall, F1 Score

In order to mitigate the shortcomings of accuracy, precision and recall or the F1-score can be used. This encompasses the especially the application with imbalanced data sets [75]. Precision is a measure of relevance, while recall is a measure of how many truly relevant results are returned. The F1 score can be interpreted as a harmonic mean between precision and recall and is commonly used as it combines the two metrics into one [80;68].

Because precision and recall are originally defined for a binary classification problem, an extended version of the metrics is required for a multi-class classification problem. Mainly two methods exists. First, the macro-averaged form, in which the metrics are assessed in a one-against-all approach for each class, followed by taking the mean over the partial results [75]. Second the weighted average of the metrics. This form works similarly, but the distribution of the classes is also taken into account to avoid overestimating the results in an unbalanced data set [80]. To have an alternative to accuracy in experiments with unequally distributed classes, only the weighted form of the metrics is considered for the experiments of this project. Formally the three metrics are defined by [81]:

Precision

$$Weighted\ Precision = \frac{\sum_{i=1}^{n} |y_i| \frac{TP_i}{TP_i+FP_i}}{\sum_{i=1}^{n} |y_i|}$$

Recall

$$Weighted\ Recall = \frac{\sum_{i=1}^{n} |y_i| \frac{TP_i}{TP_i+FN_i}}{\sum_{i=1}^{n} |y_i|}$$

F1 Score

$F1_i = 2 \times \frac{PRE_i \times REC_i}{PRE_i + REC_i}$

With $F1_i$ = F1 Score for class i, PRE_i = Precision for class i and REC_i = Recall class i

$Weighted\ F1\ Score = \frac{\sum_{i=1}^{n} |y_i| \times F1_i}{\sum_{i=1}^{n} |y_i|}$

2.5.3 Confusion Matrix

In machine learning, the confusion matrix is typically used to evaluate or visualize the performance of a model in a supervised classification problem [82;83]. Especially in a multi-class environment, the confusion matrix can help to understand the performance of the model on different classes and allows to identify classes on which the model performs well and on which it discriminates poorly (is confused) [82]. This is fairly impossible by only considering a single metric such as accuracy or much more expensive to study precision and recall for every single class. Figure 2.2 shows an example of a confusion matrix used in this project. Important to mention is, that for all confusion matrices computed in this work, the predicted labels are represented by the x-axis, whereas the true labels lie on the y-axis.

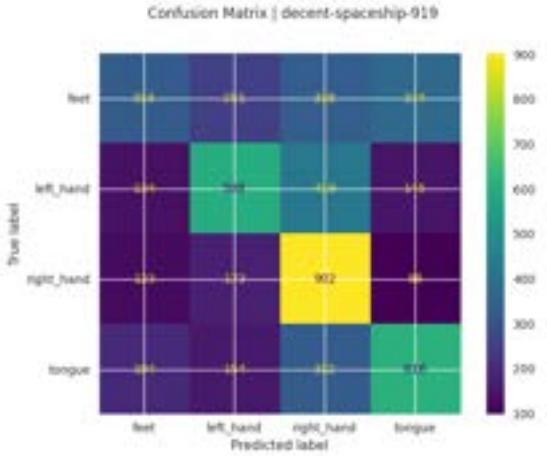


Figure 2.2: Sample Confusion Matrix

2.5.4 Receiver Operating Characteristic Curves

Computing the Receiver Operating Characteristic (ROC) curve for this work can have two main advantages. First, it provides a supplementary visualization on how well a model performs [84]. Like the confusion matrix, in a multi-class data set it provides a simple way to visually interpret the model performance on the different classes [85;84]. For all classes, the curves are calculated with a one-against-all approach as the method originally was developed for a binary classification task [84]. The curve for each class is created by plotting the true-positive-rate (TPR) against the false-positive-rate (FPR) at various threshold settings [85;84]

Second, it offers the area-under-the-curve (AUC) score as additional metric to accuracy and F1 score [84]. The area under the curve assesses the discriminative ability of a model [84]. An AUC value of 1 means a perfect and a value of 0.5 means no discriminative ability of the model [84]. Figure 2.3 shows a sample ROC curve used in this project.

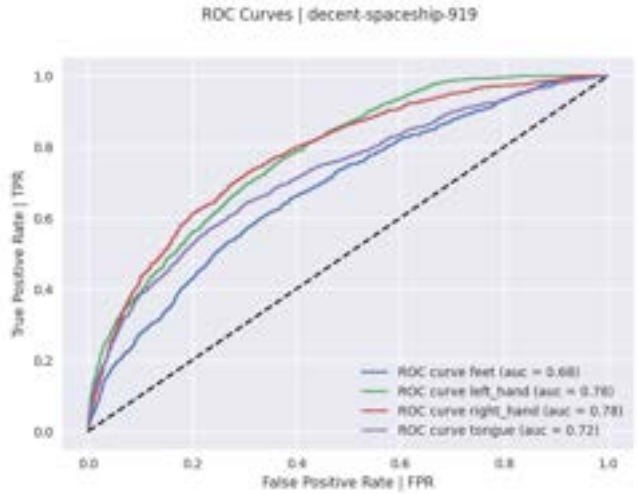


Figure 2.3: Sample ROC Curve

2.5.5 Cohen's kappa Score

Cohen's kappa score is widely applied in statistics, psychology, biology and medicine, but got only little attention in the field of machine learning [86]. Because it is used in medicine, the Cohen's kappa score is more commonly utilized in EEG classification than in other machine learning tasks [87;88]. Basically, Cohen's kappa score measures the agreement between two raters [89]. That is why the score is often considered when comparing a system with human performance [86]. When used as classification metric, the first rater are the true labels and the second rater is the classifier [87]. The score is then defined as follows [86]:

$K = \frac{P_o - P_c}{1 - P_c}$

With P_o = Probability of agreement P_c = Probability due to random chance / agreement

Cohen's kappa score is generally valued as more robust in terms of reliability and validity in comparison to the accuracy score [88]. This is especially the case when dealing with an imbalanced data set. An often referred downside of Cohen's kappa score is the harder interpretability than accuracy due to the fact that it embraces a possible value range from [-1 ; 1] [87;86]. Hence, in favour of simplicity often just accuracy is used.

2.6 Machine Learning Models

This section aims to introduce the models used during the experiments.

2.6.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are widely used on EEG data sets [27]. CNNs use learned convolutional filters to extract features for classification and are therefore often preferred over classical machine learning models because less preprocessing and expert knowledge is required [14;1].

Similar to ordinary Multi Layer Perceptrons (MLP), the representational capacity of CNNs can be scaled by increasing the number of layers. CNNs with multiple layers are called deep CNNs, those with a small number of layers are called shallow CNNs. Investigations showed that earlier layers of a CNN extract more low level features such as edges in images and later layers extract more high level features like an entire house [90]. The higher capacity of deep CNNs comes at the cost of requiring more data to

train the increased number of parameters, training may take longer and more parameters need to be stored [36].

As input, the two-dimensional EEG signal can be passed directly to the model and does not have to be reduced to a single dimension as with MLPs. This because usually two dimensional convolutional filters are used for CNNs [36]. Depending on the architecture, in a supervised classification problem, the extracted and flattened features are then passed to one or more linear layers, followed by a Softmax layer to discriminate between the different classes. The output of the model is a probability distribution over the classes of an EEG data set.

In the literature, several architectures of CNNs for motor imagery data sets are proposed [36;91;92;35]. This work compares two different CNN architectures which are explained in more detail in the following sections.

2.6.2 Shallow Convolutional Neural Network

The first architecture used is a shallow CNN introduced by Schirrmeister et al. [36], which is visualised in figure 2.4.

The first two layers of the CNN comprise a temporal and a spatial convolutional filter, similar to a Filter Bank Common Spatial Filter (FBCSP) approach [36]. FBCSP is a combination of a filter bank (FB) together with the Common Spatial Patterns (CSP) algorithm [93]. Basically, a filter bank is a set of bandpass filters that separates the input signal into several components that represent a frequency sub-band of the original signal [93]. For each sub-band, the CSP algorithm then tries to find a set of spatial filters that maximise the difference between two classes of a data set [94]. But in contrast to FBCSP, the CNN combines both computational steps in one network and there is no need to define frequency bands by hand [36].

After the first two layers, a non-linear activation function is plugged in, followed by a mean pooling layer. According to Schirrmeister et al. [36], the pooling layer enables the network to learn a temporal structure of the band power changes within a trial. Finally, a linear classification layer is used together with a softmax activation function to obtain a probability distribution over the classes of the data set [36].

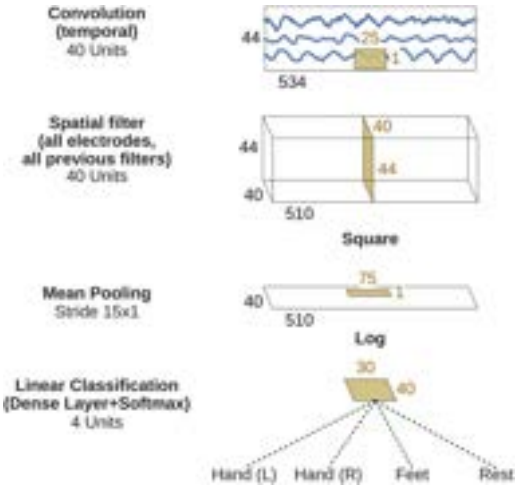


Figure 2.4: Architecture Shallow CNN
[36]

2.6.3 Deep Convolutional Neural Network

To contrast the results of the shallow CNN, this project also conducts experiments with a deeper CNN proposed in the work of Schirrmeister et al.[36]. Instead of two convolutional layers in the shallow network, the architecture of the deep CNN consists of four convolutional layers, as visualised in figure 2.5.

Similar to the shallow network, the first two layers represent a temporal and a spatial convolutional layer, followed by a max pooling layer. The max pooling layer is used in order to reduce the dimensionality and make the system more transformation invariant[95]. In the original paper, no clear evidence was found why maximum pooling was used for the deep CNN and average pooling for the shallow network and rather based on empirical testing[36]. A possible explanation could be that peaks in the signal should be preserved to deeper layers of the network, while the shallow CNN relies more on trends in the signal that are smoothed by mean pooling. Three additional convolutional blocks consisting of a convolutional and a max pooling layer complete the feature extraction layers of the network. The increased number of layers should lead to higher level representations of the signal as input features for classification in comparison to the shallow network[14]. Finally, a linear layer is used to compute the probabilities to a set of classes given by a data set.

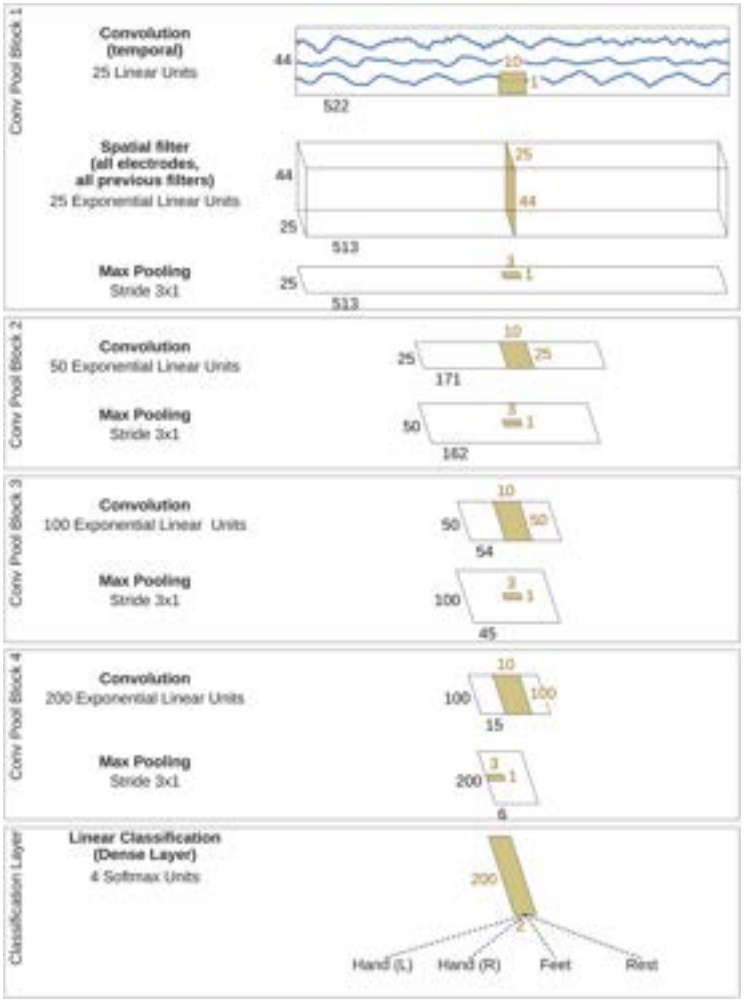


Figure 2.5: Architecture Deep CNN
[36]

2.6.4 Support Vector Machine

To get a first impression about the effectiveness of augmentation techniques for classical machine learning algorithms for motor imagery data, a support vector machine (SVM) is used as third model of this project.

Many classical models such as K-Nearest-Neighbour (KNN), Support-Vector-Machine (SMV), Linear-Dicriminant-Analysis (LDA) and Random Forests exists. In the literature, the SVM mostly dominates with the best average performance on motor imagery classification tasks [96;97;98;99;100]. One explanation noted in the literature is that the SVM can better deal with high dimensional data and is therefore more resistant to the curse of dimensionality [100].

A SVM is a type of supervised learning algorithm that can be used for classification or regression tasks and was originally introduced by Cortes and Vapnik [101]. Since then, SVMs have been applied successfully in various areas, including economics, bioinformatics and motor imagery BCI systems [102;97]. It works by finding the hyperplane in a high-dimensional space that maximally separates the different classes [103]. SVMs use kernel functions to map an originally non-linear separable feature space into a higher dimensional Hilbert space in which the data is linearly separable [102].

Many hybrid combinations of SVM with other techniques are also mentioned in the literature for motor imagery classification tasks. Dong et al. [104] use a hierarchical classification approach by leveraging a hierarchical SVM (HSVM) to improve classification performance. While Ma et al. [99] propose a particle swarm optimisation (PSO-SVM), Selim et al. [105] apply a Bat algorithm (BA) in a CSP-BA-SVM for an optimised hyperparameter tuning instead of using grid search. For simplicity, this work uses the default implementation of SVM with different kernel functions.

2.7 Augmentation Techniques

Data augmentation is a widely used strategy if data is scarce, the data is imbalanced or if there is explainable variance within the data which can be generated synthetically [18]. Moreover data augmentation techniques are often used to mitigate the problem of overfitting [18]. However, data augmentation should only be used if it is unfeasible to gather alike samples from the real data distribution [18]. The main idea behind data augmentation is to apply label preserving transformations to an input data set in order to add more invariant examples [106].

Formally, to a sample-label pair (x, y) a transformation function $\Phi(x)$ with a certain probability p_{aug} is applied *a priori* [27]. If a model is trained accordingly, it should theoretically output the same probability distribution for the augmented input $P(y|\Phi(x))$ as for the same input sample without the applied transformation $P(y|x)$ [27].

Augmentation techniques can be applied on the different domains of an input signal [35]. There exist strategies for the time, the frequency and the spatial domain [27]. In this report we use a mix of techniques of which each targets one of this domains.

In order to be able to conduct augmentation experiments on EEG data, strategies need to be selected. Several augmentation strategy investigations and experiments were conducted in the literature [28;29;32;33;27]. Also diverse augmentation strategies for EEG data as well as for motor imagery tasks were proposed [19;20;21;22;23;24;25;26]. This section aims to discuss applicable augmentation strategies from the introduced papers, which had a positive effect on classification performance of a motor imagery classification task and are implementable in the scope of this work.

2.7.1 Frequency Shift

Frequency shift is an augmentation technique which targets the frequency domain of an input signal. It was first introduced to mitigate the problem of inter-subject variabilities in sleep stage classification, where changes in the frequency of a signal indicate a stage transition [27]. According to Rosenberg and Van Hout dominant and characteristic peaks in the frequency of EEG signals can be vital for a model to be able to distinguish between classes [107]. The experiments of Rommel et al. [27] also showed that it can be beneficial to use this augmentation technique for a motor imagery task in a cross-session or inter-subject setting. This because it was examined that the shape of the signal between subjects can be likewise but the peaks within a window could occur at a different frequency [27]. Since the experiments conducted are based on cross-session and inter-subject settings, it could be helpful to use this method.

The technique described by Schirrmeister et al. [36] and Rommel et al. [27] allows to specify a maximum magnitude Δf by which the power spectral density (PSD) of the signal gets shifted within a window. The final magnitude applied then gets sampled from a uniform distribution between the interval $[-\Delta f_{max}, +\Delta f_{max}]$ in order to prevent a shift of the mean within the training data [35;27]. Additionally, the probability can be defined to which this transformation is applied to an input window. An example of shift of $+\Delta f = 2\,Hz$ is shown in figure 2.6

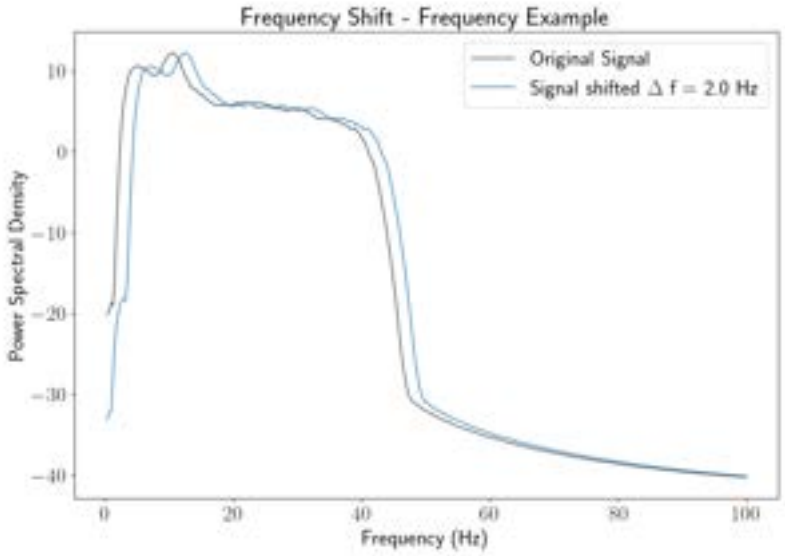


Figure 2.6: Frequency Shift - Frequency Example
[36]

2.7.2 Fourier Transform Surrogate

The aim of the Fourier Transform Surrogate (FT Surrogate or FTS) method is to generate a different waveform of the input signal while leaving the PSD unchanged [19]. Using FT Surrogate as EEG augmentation technique was first proposed by Schwabedal et al. [19] to tackle class imbalances in EEG sleep staging data. Generally, this method decomposes the Fourier components s_n of a signal x_n into amplitudes a_n and phases Φ_n. The method assumes that the EEG signals are generated by *stationary linear processes* [19] in which the Fourier phases are random numbers in the interval $[0, 2\pi)$. For example, the new random numbers can be taken uniformly from the interval $[0, \lambda \cdot 2 \cdot \pi)$, while leaving the amplitudes untouched [27;36]. Lambda describes the phase noise magnitude which is a hyperparameter and can take a value between zero an one. Lastly, the original phases of s_n are replaced by the new sampled ones and transformed back to the time domain by a inverse Fourier transform [19]. Like this,

a statistically independent sequence x_{t_n} is drown while representing the same generating distribution and preserving the frequency-band power ratios of x_n [19].

The underlying idea of this augmentation method is to force the model to focus on the PSD rather than the signal's time representation [27]. One the one hand, this could especially be helpful in a motor imagery tasks where the time on which a change in the signal occurs is not the most relevant information. One the other hand this method should be used careful on every task where the information of the time domain is relevant. Figure 2.7 shows an example of an original representation of a signal and by one of it's FT Surrogates. It can be seen that important peaks at a given time point vanished in the new representation. In addition, it was examined in the figure 2.8 whether the spectral representation remained the same before and after the transformation. It can be seen, that the band-power ratios remained the same, but the entire PSD function gets slightly squished and smoothed.

FT Surrogate has two degrees of freedom, the phase noise magnitude λ and the probability p_aug with which the technique is applied to a certain trial.

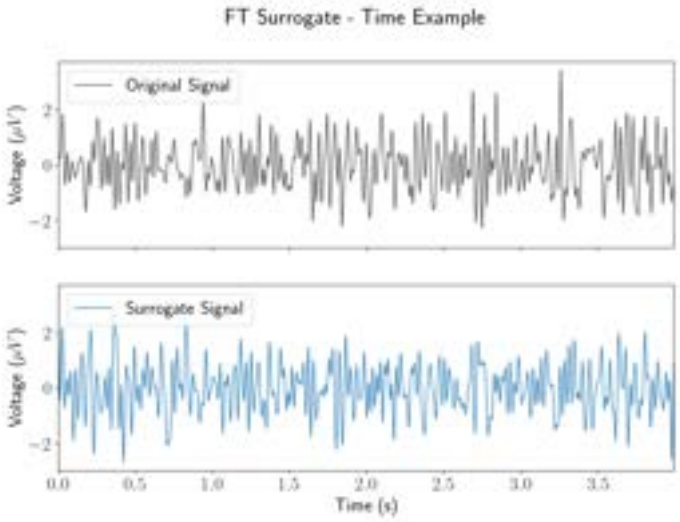


Figure 2.7: FT Surrogate - Time Example
[36]

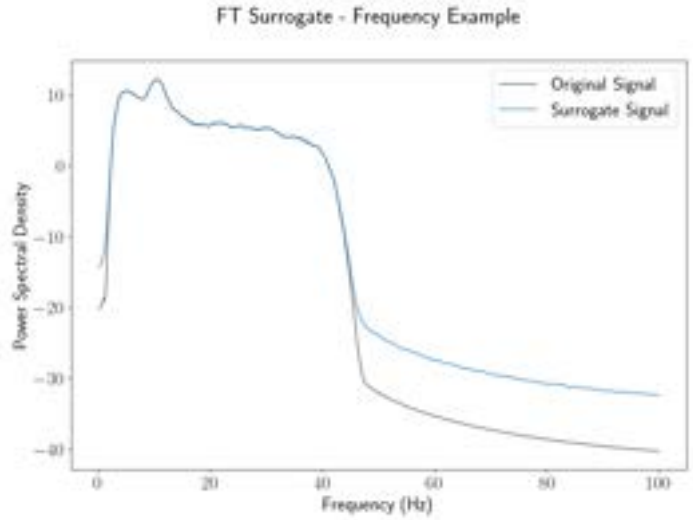


Figure 2.8: FT Surrogate - Frequency Example
[36]

2.7.3 Channels Dropout

The Channels Dropout is an augmentation technique in the spatial domain of an EEG signal [27]. Like the original dropout layer [108], where a neuron of a dense layer is randomly omitted, the idea of the Channels Dropout is to set the amplitudes of a randomly picked EEG channel to zero [20]. In other words, the information of a channel gets randomly removed by a defined probability p_{drop} [20].

This augmentation technique was first introduced by Saeed et al. [20] with the idea to reduce the model's dependency on a given input channel. As a result the model should be more error prone against overfitting and show a better generalization. Regarding Banville et al. [109] the lack of a consistent quality of the input signal recorded by a dedicated channel is one of the major challenges when dealing with EEG signals. Moreover, EEG channels are highly correlated and often have a low signal to noise ratio [1]. With Channels Dropout, the model should get invariant to the number of input channels and learn on the entire information available, instead of focusing on single channels [27].

Formally, the technique is setting the values of a channel to zero for each input window $X \in \mathbb{R}^{C \times T}$, where C denotes the number of channels and T the number of samples over a given time [27]. Finally, the augmented signal has the form $ChannelsDropout[X]_c := d_c \cdot X_c$, where d_c is a vector of length C and consists of values of a Bernoulli distribution $\mathbf{B}$ with a probability of p_{drop}. As the formal definition suggests, the only parameter that can be tuned with this augmentation technique is p_{drop}.

2.7.4 Sign Flip

While it is important for augmentation techniques to preserve the validity of a given input, it is not obvious why a flip of the sign of an EEG signal can fulfil this requirement. In the literature, Freer et al. [35] note that the signal retains its frequency, spatial and power components even when the sign is inverted, making this method a valid augmentation technique for EEG data. The method looks similar to a phase shift of 180 degrees. However, with sign flip the timing of the oscillations remains the same in contrast to a 180 degree phase shift where the timing of the signal's oscillations is reversed [35;110].

Additionally, Rommel et al. [111] argue that the information encoded in the electrical potentials captured by EEG sensors are likely to be invariant to the polarity of the electric field. The polarity of an EEG signal refers to direction of the flow of electric charges along neuron dendrites [111;112;113]. Since charges moving to deeper layers of the cortex as well as moving up to superficial ones, both signs are likely to occur. Finally, Rommel et al. [111] also point out that the polarity can also be influenced by the choice of reference electrode.

The sign flip augmentation method is defined by multiplying each channel by -1 with a probability of p_{aug} which results in $SignFlip[X](t) := -X(t)$. Ultimately, the only hyperparameter to be tuned for the sign flip augmentation technique is p_{aug}. Figure 2.9 shows an example of an applied sign flip to the signal of a given input channel.

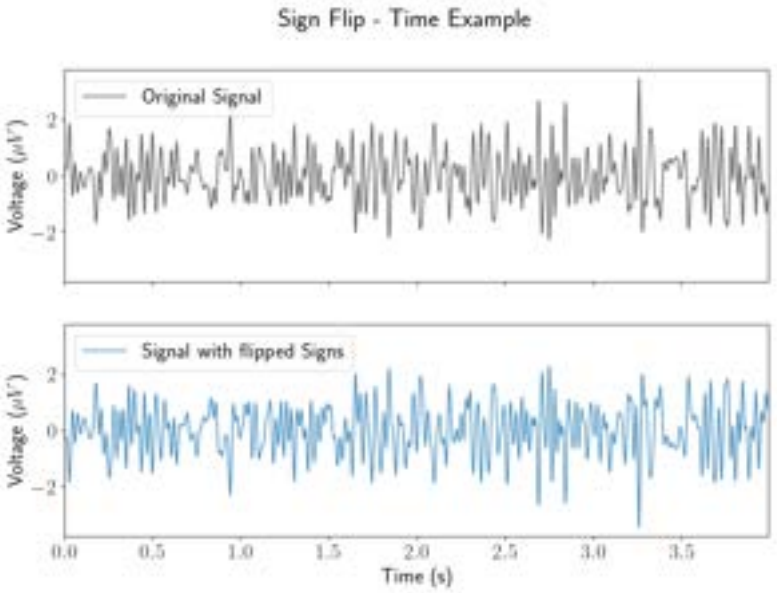


Figure 2.9: Sign Flip - Time Example
[36]

2.7.5 Time Reverse

Reversing the time of an EEG signal has no effect on the power components of a signal while generating a new input[35;27]. Nevertheless, it can be assumed that changing the time domain of an input signal can be misleading if there lies important information in the sequence of events happening during an epoch. In motor imagery, where a significant part of the information is contained in the frequency range of the EEG signal, this augmentation method could still be helpful by generating additional input signals. Furthermore, it would be beneficial if a model could become time-invariant to when a particular pattern or task occurs in a certain time window, which could eventually increase a model's generalisation capabilities[27].

The technique is defined by inverting the time axis t with $TimeReverse[X](t) := X(t_{max} - t)$ where t_{max} is the length of a time window. As Figure 2.10 shows, the method can also be described as a vertical flip of an input channel and has therefore some similarities to the sign flip technique. The only hyperparameter also for this strategy is therefore the probability p_{aug} with which the transformation is applied to a given channel.

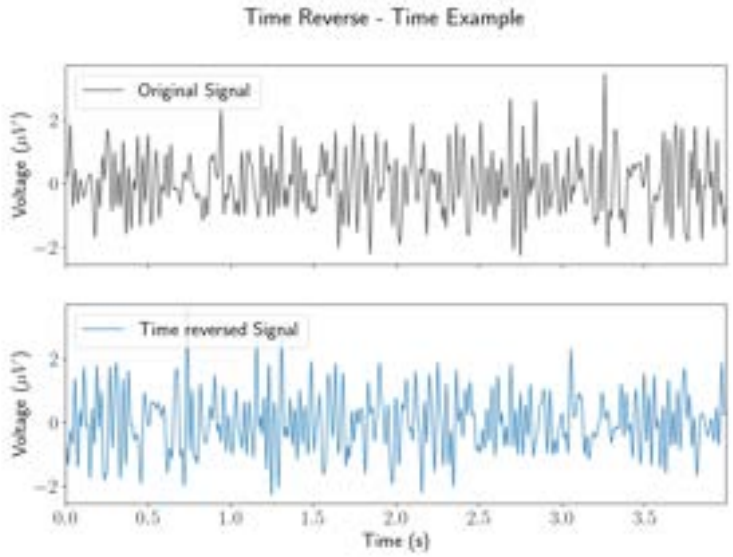


Figure 2.10: Time Reverse - Time Example
[36]

2.7.6 Smooth Time Mask

Masking of input data is used as augmentation technique in various domains[114;115]. It means that a certain area of an input is replaced by an other signal. For the EEG data in this work Smooth Time Mask (STM) is applied on the time domain, whereas it would theoretically also be possible to use it in the frequency domain. The method proposed by Cheng et al.[24] consists of randomly replacing a certain time frame of an input window by another substitute input. There are multiple variants of signals that can form substitutes[24]. For instance, random noise from a Gaussian distribution or it could simply be replaced by zeros, as used for this work[23;24]. Depending on the substitute chosen, the difficulty for the model to generate a valid representation changes, which is why this method can also serve as a regularisation technique[24]. Beside the substitution type, the three degrees of freedom of STM are the length of mask, the position of the mask and the probability p_{aug} with which transformation is applied to a given window[23].

The idea of this augmentation technique is to reduce the model's dependence on a certain parts and transient patterns of an input window[27]. Figure 2.11 shows an applied zero-mask of one second in the middle of an input window of four seconds.

In addition, to examine the effect on the frequency domain, figure 2.12 shows the PSD before and after applying Smooth Time Mask to the same signal. The visualisation shows, that some information in the frequency domain gets lost, by smoothing out the curve for some frequency bands. If the curve is smoothed, this could lead either to less variance between subjects, or important features for the model to classify the signal were removed. There is also a reduction in PSD in some frequency bands, which leads to a vertical shift of the signal in that band.

Formally, the technique is applied by replacing a proportion Δt that starts from a randomly sampled time point t_{cut} within a given window[27]. The computation is then conducted by multiplying the input signal X by a mask m_λ, which is obtained by the result of two opposing sigmoid functions of temperature λ[27]. The method is then defined by the following formula and its components[27]:

$SmoothTimeMask[X](t) := X(t) \cdot m_\lambda(t)$ with
$m_\lambda := \sigma_\lambda(t - t_{cut}) + \sigma_\lambda(t_{cut} + \Delta t - t)$
$\sigma_\lambda(t) := \frac{1}{1+exp(-\lambda t)}$
$t_{cut} \sim \mathbf{U}[t_{min}, t_{max} - \Delta t]$

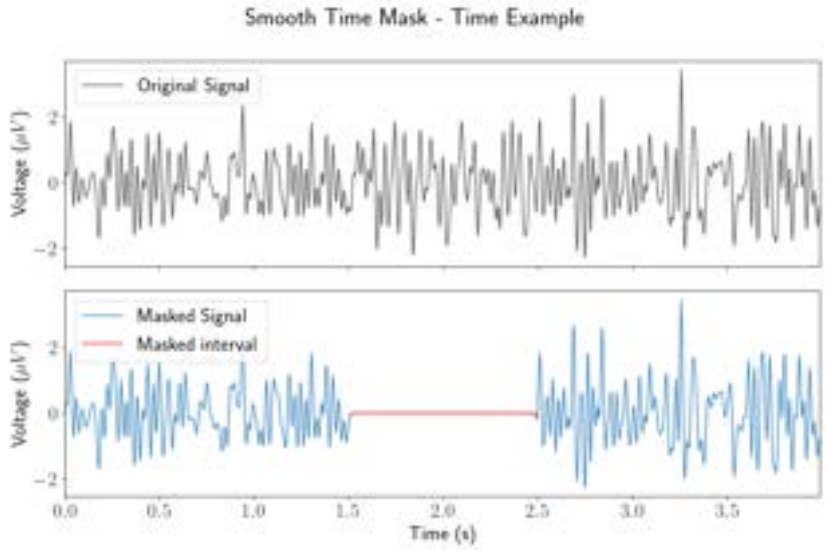


Figure 2.11: Smooth Time Mask - Time Example
[36]

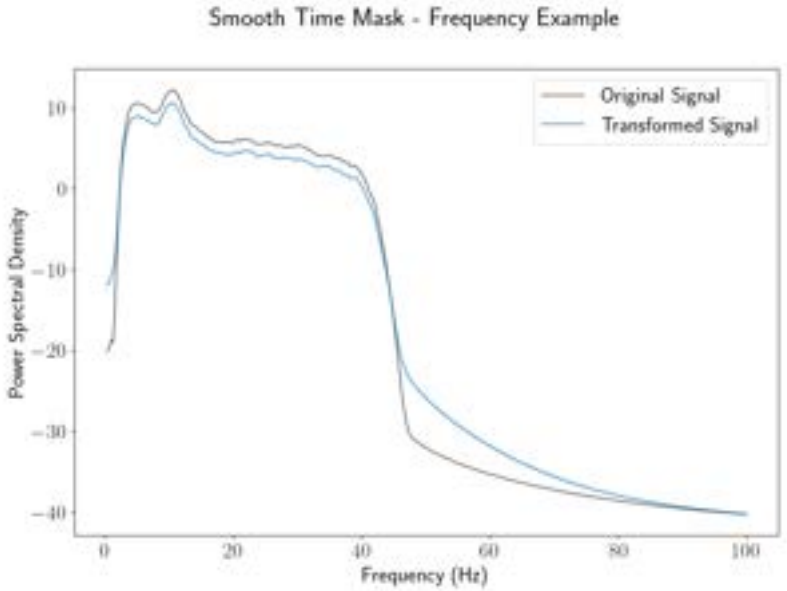


Figure 2.12: Smooth Time Mask - Frequency Example
[36]

Chapter 3

Methods

3.1 Training Scenario Selection

This section aims to describe the training scenarios used in this project. As a result, data sets can be selected and the applied partitioning strategy elaborated 3.2.

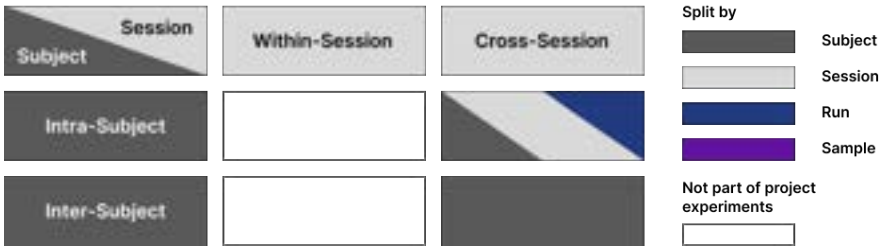


Figure 3.1: Training Scenario Selection

As described in figure 3.1, this project includes experiments on cross-session classification strategies for a single subject (intra-subject) as well as for multiple subjects (inter-subject). This because for many strategies, it is still unknown, if applied augmentation strategies achieve a divergent magnitude of change in classification performance in different training scenarios [27]. Due to the constrained scope of this work, not all strategy combination can be explored. Those two combinations were chosen because of the higher benefit or applicability of those strategies in real world applications.

Depending on the prevalent structure in a dedicated data set, the implementation of the selected strategies can vary. This section should give details about the general implementation of the strategies over all data sets, whereas the granular descriptions of the strategy applications on a particular data set can be found in the specific introduction in section 3.2.

3.1.1 Intra-Subject

In order to prevent a premature conclusion of the performance reached in an intra-subject setting, each experiment is conducted for each subject independently. Subsequently, the mean over all subjects is taken for the final assessment of an experiment's performance. This method also allows to display and to reason about the result distribution between subjects and also helps to draw a conclusion about how sensitive a model reacts on the inputs given by specific subjects.

In this training scenario, because of the absence of multiple subjects, different sessions can be taken to split between training, valid and test set. If only a limited number of sessions is available in the data set,

valid and test set can also be split by runs from the same session. However, this should be conducted carefully to not end up in a double dipping problem and no runs of the training session should be used in valid or test set.

With the implementation of this multi individual subject analysis, a cross-validation (CV) like evaluation is accomplished for the intra-subject strategy.

3.1.2 Inter-Subject

Similar to the intra-subject strategy, a cross-validation like evaluation of the performance metrics needs to be achieved in order to draw a data set partitioning independent conclusion. Nevertheless, in comparison to the intra-subject implementation, the cross-validation in the inter-subject setting of this project is reached by first split up the data set by subjects. For the test set, a static number of subjects is separated for the final validation depending on the data set size. The remaining data set is then divided into a train and validation set in a leave-one-out-cross-validation (LOOCV) manner. This means that one single subject per run is used as validation set and all other subjects are used for training. After one run, the validation subject is replaced by another until each subject of the remaining data set (after train-test-split) has been used once as a validation set. Also with this approach, it is possible to evaluate an average performance. Finally, it increases the robustness of the evaluated training and validation metric as it is independent of the partitioning of the data set.

3.2 Data Set Selection and Split Strategy Application

Based on the decisions made in the previous section 3.1, the data sets for this project can be selected. Because the scope and time constraints do not allow to collect self-recorded data sets, publicly available EEG data sets need to be found and analysed. In addition, conducting experiments on public available data sets increases the probability to get comparable results by other researchers and allow them to reproduce the findings, when a own recorded data set cannot be published. To cohere with the training scenarios of this project, the public data sets need to meet the following requirements:

- the data set must encompass multiple subjects
- the data set must encompass multiple sessions
- it is favoured that the data set consists of multiple runs in order to enable follow up experiments or having a larger data volume at hand if required.

3.2.1 Data Set Source

There are several EEG data sets publicly available. For this project, data sets distributed via «Mother of all BCI Benchmarks» (MOABB) `https://neurotechx.github.io/moabb/` were considered[116]. At the time of the report, 17 publicly available motor imagery data sets were accessible via the API of MOABB, ranging from competition data to data sets published as part of publications[116]. For the experiments of this work, two of them were selected according to the defined requirements in section 3.2.

3.2.2 Data Set 1 | BNCI2014001

The BNCI2014001 data set origins out of the BCI Competition IV[117]. The data set consists of EEG data from nine subjects and two sessions recorded on different days[117]. The subjects were asked to perform four different motor imagery task, namely the imagination of the movement of the left hand (class 1), right hand (class 2), both feet (class 3), and tongue (class 4)[50]. Each session is comprised of six runs separated by short breaks. One run consists of 48 trials (12 for each of the four possible classes),

yielding a total of 288 trials per session [50]. The trials with a duration of four seconds were sampled with 250 Hz and bandpass-filtered between 0.5 Hz and 100 Hz, recorded by 22 EEG electrodes [117]. The sensitivity of the amplifier was set to 100 μV and a 50 Hz notch filter was applied to suppress input signals considered as noise by the creators of the data set [50]. Figure 3.2 shows the split strategies for the BNCI2014001 data set specific to an intra- and inter-subject training scenario.

As described in section 3.1, each experiment gets conducted for each subject individually and sequentially when applying an intra-subject strategy. During each processing of an individual subject, the first session is used as training set. Because of the availability of runs in this data set, a split of the second session by runs is used as validation and test set.

In contrast for the inter-subject training scenario, subject four is used as test subject. All remaining subjects are partitioned in a LOOCV fashion for either train or validation set. Because the partitioning here only depends on the subjects, both sessions were used for training, validating and testing.

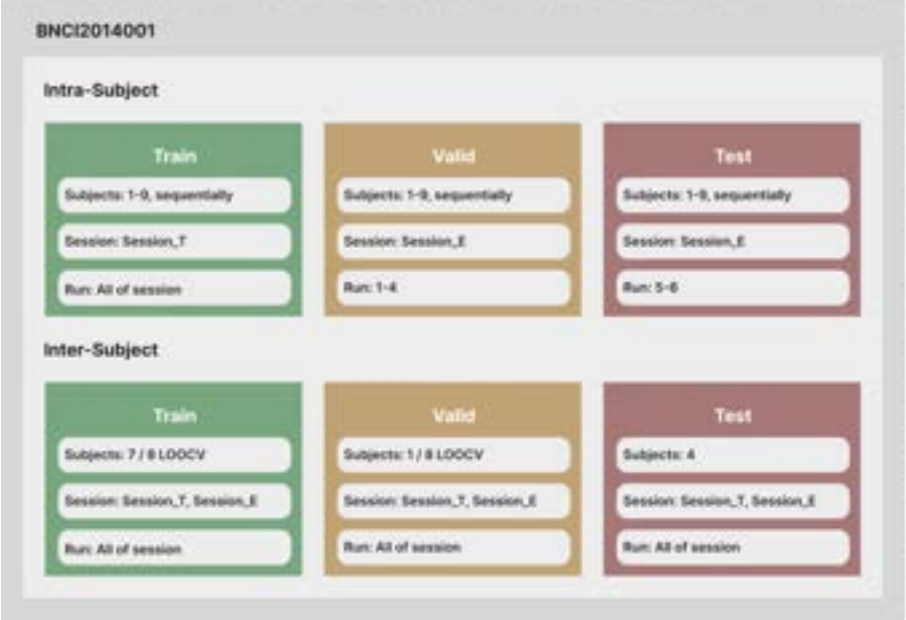


Figure 3.2: Data Set BNCI2014001 Split Strategy

3.2.3 Data Set 2 | BNCI2014004

The motor imagery BNCI2014004 data set was created and used in the 2008 BCI competition [50]. In the original paper of Leeb et al. [37], the data set consisted of ten subjects, which were asked to perform the imagination of left hand (class 1) and right hand (class 2) movements. However, the actual data set accessible via MOABB only contains nine subjects of which all were right handed [116]. For each subject, five sessions are provided in the data set, whereby the first two sessions were recorded without, and the last three sessions with feedback [37;116]. In the original paper only two sessions were mentioned, one for screening and one for feedback [37]. Nevertheless, those two sessions were recorded on two different days within two weeks [37]. Every session embodies six runs of twenty samples, making up a total of 120 balanced trials (60 class 1, 60 class 2) per session with a duration of 4.5 seconds per trial [37]. Although the data was recorded in different runs, in the public accessible data set of MOABB, all runs were tied together to a single run per session and not further annotated [116]. The data was gathered by using three bipolar EEG sensors (C3, Cz, and C4) with a sample frequency of 250 Hz and an applied band pass filter between 0.5 Hz and 100 Hz [37].

This data set was selected due to its binary classes in order to facilitate a comparison between binary and multi-class classification problems.

Figure 3.3 shows the used split strategy of the BNCI2014001 data set for an intra- and inter-subject training scenario. In comparison to the BNCI2014001 data set, the BNCI2014004 data set does not comprise runs but has multiple sessions. This is why for the intra-subject strategy session one to three is used as training set, session four as validation set and session five as test set.

For the inter-subject training scenario similar splits are applied as for the BNCI2014001 data set.

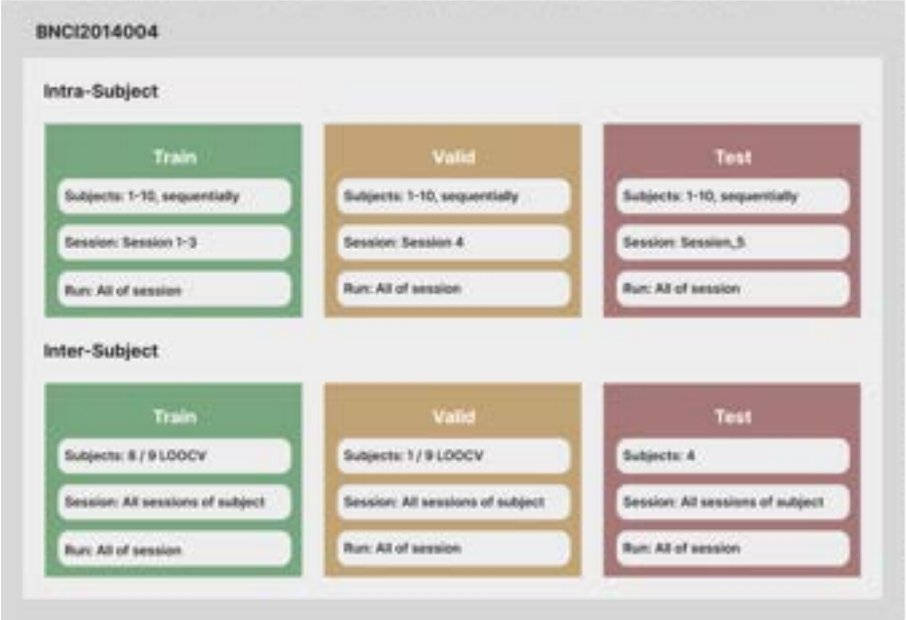


Figure 3.3: Data Set BNCI2014004 Split Strategy

3.3 Preprocessing

In the experiment pipelines of this work, only minimal preprocessing steps are applied. The relevant features should rather be learned by the convolutional neural networks during training. Because tuning and exploring on further preprocessing techniques would go beyond of the scope of this work, the methods and parameters suggested by Schirrmeister et al. [36] were used for this project.

The preprocessing before initialising the model is divided into four different steps. First the intended channels of a data set are selected. Some data sets on MOABB not only comprise EEG but also Magnetoencephalography (MEG) or Stimulation (STIM) Electrodes. For the experiments in this work, only the EEG signal is used, which is why all other channels are discarded in the first preprocessing step. Secondly, the the data is scaled to transform amplitudes from V to μV. As third step, a band-pass filter is applied to remove avoidable noise. The analysed data gets high passed filtered above 4 Hz and low pass filtered below a 38 Hz cut off.

Finally, before initialising the model, a exponential moving standardisation is performed in order to scale the input signals into a proper regime for the activation functions, which speeds up learning and mitigates the risk of vanishing or exploding gradients. In contrast to usual standardization, mean and standard deviation are only calculated for a certain window and recalculated as the window roles over the signal. This is beneficial for non stationary signals like EEG.

For this project, exponential moving standardisation is performed as follows: As first step, the exponential moving mean m_t at time t is computed by $m_t(x_t) = \lambda \cdot mean(x_t) + (1-\lambda) \cdot m_{t-1}$ [36]. Here, λ represents the decay factor [36]. It controls how much weight is given to more recent data points [36].

Then the exponential moving variance is computed by $v_t(x_t) = \lambda \cdot (m_t - x_t)^2 + (1-\lambda) \cdot v_{t-1}$. Eventually, the standardisation is applied to a given data point x_t at time t as $x'_t = \frac{(x_t - m_t)}{max(\sqrt{v_t}, \epsilon)}$, where ϵ is used to prevent the division by zero.

As last preprocessing step after initialising the model, a distinct trial is cropped into several windows to increase data efficiency. In order to decide on the parameters required for the window cropping, information of the initialised model is required. Namely, the final convolutional receptive field size to decide on the window stride during cropping.

A summary of the preprocessing applied and the used hyperparameters in the implementation can be found in appendix A.2.1.

3.4 Metric Selection

An important task of any data science project that involves training and evaluating specific models is to establish an appropriate and meaningful performance metric. Several metrics and methods were introduced in section 2.5.

As suggested by Ng[68], a single metric is used to compare the different experiments of this work. For that, accuracy was chosen because of its ability to provide an efficient and easy-to-interpret indication of changes in classification performance. More precisely, not the absolute accuracy values are used, rather the relative change in percent if a certain augmentation technique is applied. In this project, accuracy does not suffer from the limitations discussed in chapter 2.5 because of the evenly balanced data sets. However, all of the presented metrics are computed and stored for all experiments if further analysis is required.

In addition to the number-based metrics, performance charts are created for all experiments. Namely, a multi-class confusion matrix and ROC curves with the corresponding AUC values for all classes in the data set. With this approach, not only the classification accuracy of a model can be evaluated, but also the differences between and the sensitivity to classes can be investigated.

3.5 Augmentation Technique Selection

Because of the scope of this work, the number of applied augmentation techniques in the experiments needs to be limited to three methods. The techniques introduced in section 2.7 were used in previous publications. When selecting the methods for the experiments in this work, experience from previous work should be employed. Therefore, table 3.1 shows the results obtained by Rommel et al.[27]. The table also shows the dedicated hyperparameter values applied to obtain the listed performance. The techniques for this project were selected based on the highest increase in relative accuracy when applying a particular technique to a motor imagery data set compared to using the same model without applying the technique. As a result, Smooth Time Mask, Channels Dropout and FT Surrogate are the selected augmentation strategies further evaluated in the experiments of this work.

Technique	Tuned Hyperparameter (HP)	HP value	Increase Accuracy in %
Smooth Time Mask	Δt	1.6 s	∼24%
Channels Droput	p_{aug}	1	∼21%
FT Surrogate	Δ Φ max	9/10 π rad	∼18%
Time Reverse	p_{aug}	-	∼8%
Frequency Shift	Δf	2.7 Hz	∼3%
Sign Flip	p_{aug}	-	∼0%

Table 3.1: Relative Increase of Accuracy Depending on Applied Augmentation Technique in Literature [27]

3.6 Experiments

The experiments of this work are divided into two main parts. First, a baseline tuning is conducted for each model and every data set. Second, experiments with the proposed augmentation strategies are performed to see the effect on the performance in comparison to the obtained baseline. This section aims to give additional details on the two types of experiments carried out in this report. Especially about the structure, the assumptions and scope decisions made. The entire list of conducted experiments can be found in appendix A.2.2

3.6.1 Baseline Tuning

The baseline experiments are conducted to obtain a performance value that can be compared with the results received when applying the selected augmentation strategies. The baseline performance should be comparable to those found in the up to date literature in order to prevent a comparison which fails in the foundation of the model. However, it is not the aim of this work to tune a model's hyperparameter in an exhaustive way. Nevertheless, this section should show how the baseline experiments are set up, how the terms of the experiments are defined, which parameters were tuned for the considered models and what data is logged for the experiments.

Structure

A baseline experiment can be separated into the following three levels. The experiment level, the trial level and the run level. The experiment itself is defined as the top-level entity to tune a selected model for a given data set. For instance, an experiment can be the process of tuning a shallow CNN for the BNCI2014001 data set. Each experiment gets an id (e.g. BNCI2014001_Inter-Subject_NN-Shallow_Tuning_Default_None) for identification, later reference and for building the relations between the three levels.

Each experiment consists of multiple trials. A trial is a random combination out of possible hyperparameters with which a model is trained. Because of the training duration of a trail and the time scope of this work, it is not feasible to conduct an exhaustive grid search over all possible hyperparameter combinations. That is why Hydra in combination with an Optuna sweeper is used to pick the hyperparameter combinations randomly[118;119]. With Hydra all configurable parameters are set in a tree of configuration files and with Optuna ranges or choices of values can be defined, over which the framework should iterate for tuning the model. In addition, the configuration files improve the understandability of the coverage of a certain experiment, enhance the reproducibility and enables further experiments in a simple configurable way. The limit of trials per baseline experiment is set to 20 for this work.

Lastly, every trial also consists of multiple runs. The definition of a run depends on the applied split strategy. This means on one hand, for an intra-subject scenario, a run is conducted for each subject in the data set. The average performance over all individual subjects then builds the performance of a trial. One the other hand for an inter-subject scenario, a run is conducted for all training subjects with one subject using as validation set. Again, the average performances obtained with this LOOCV over the training subjects constitutes the performance of a trial. To ease the comprehensibility of a baseline experiment, figure 3.4 is visualizing the basic structure of it.

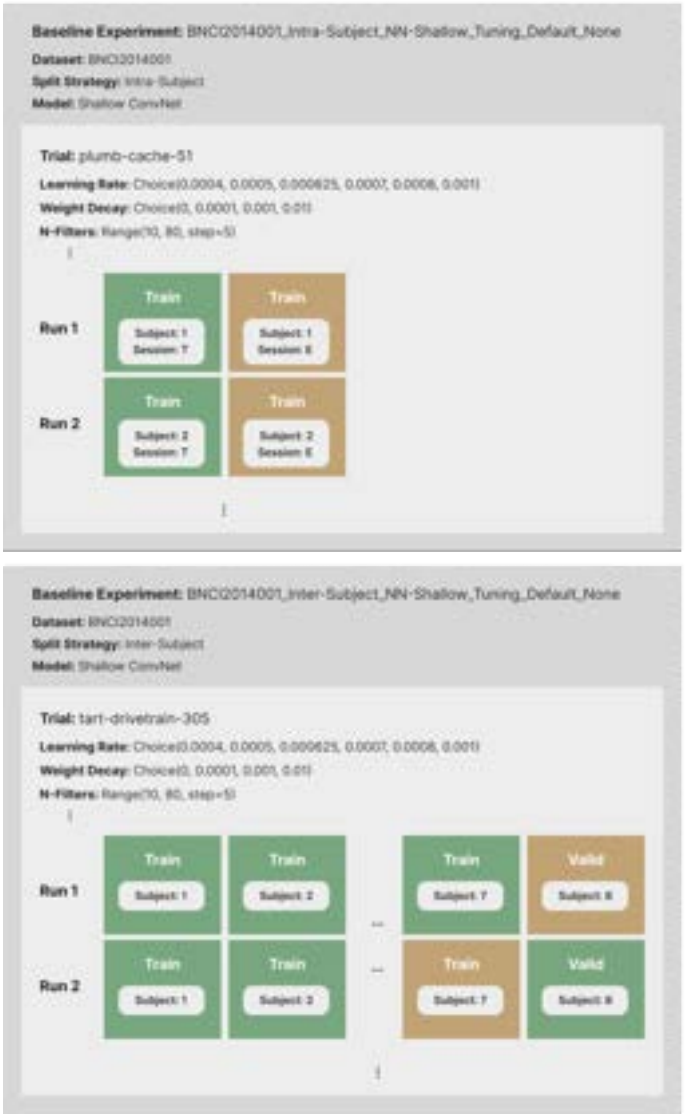


Figure 3.4: Baseline Experiment Structure

Training

Depending on the model used during the baseline experiments, a different dictionary of hyperparameter was used. The parameter options for the CNNs were identified by the use of the original implementation of Schirrmeister et al. [36]. For any numerical parameters for which no suggestion in the literature exists, a log scale parameter search was used to get a feeling for possible value range for a dedicated parameter. Table 3.2 shows the used hyperparameter search space for each of the three models.

Model	Parameter	Values
Shallow CNN	Learning Rate	*Choice(0.00001, 0.0004, 0.0005, 0.0006, 0.000625, 0.00065, 0.0007, 0.0008, 0.001, 0.01, 0.1)*
	Weight Decay	*Choice(0, 0.0001, 0.001, 0.01)*
	N-Filters	*Range(10, 80, step=5)*
	Dropout	*Range(0.1, 0.6, step=0.05)*
	Activation Function	*Choice('elu', 'relu', 'leaky_relu')*
	Batch size	*Range(16, 64, step=8)*
Deep CNN	Learning Rate	*Choice(0.0005, 0.001, 0.005, 0.008, 0.01, 0.015, 0.02, 0.03, 0.1)*
	Weight Decay	*Choice(0, 0.0001, 0.00025, 0.0005, 0.00075, 0.001, 0.01)*
	N-Filters	*Range(10, 40, step=5)*
	Dropout	*Range(0.1, 0.6, step=0.05)*
	Activation Function	*Choice('elu', 'relu', 'leaky_relu')*
	Batch size	*Range(16, 64, step=8)*
SVM	Kernel	*choice('poly', 'rbf', 'sigmoid')*
	C	*choice(0.0001, 0.001, 0.01, 0.1, 1, 1.5, 2)*
	Degree	range(1, 8, step=1)
	Gamma	*choice('scale', 'auto', 0.1, 0.001, 0.0001, 0.5, 0.05, 0.005, 0.8, 0.08, 0.008)*

Table 3.2: Tuned Hyperparameter Space Baseline Experiments

Logging

For the baseline experiments different levels of logging and result persisting are applied. First, all logs and generated plots while training are stored on the local instance. This allows valuable debugging possibilities when logging problems occur during a long run of training. Additionally the integration of Weights & Biases (WandB), a experiment tracking service, can be configured and activated for the project [120]. This has the advantage that the results are available independently of the machine on which the experiment was conducted and accessible via API for further analysis or visualization. Also, WandB is based on a document-oriented database (MongoDB) which allows to query the data in a systematic way [120]. Eventually, WandB is used to store configurations, performance metrics and plots of a conducted experiment and all its trials and runs.

The decision to use WandB as persistence layer has advantages and disadvantages. It is simple to integrate, the results are still accessible even if there are problems with the server on which the experiments are conducted and it allows to generate visualizations and control the experiments on the fly. Furthermore, additional services such as Slack can be integrated as a communication channel to get informed when an experiment has been successful or as soon as an error occurs. A disadvantage is the dependence on an external service and a online API to visualize the data. Due to the fact of a limited time span of this work, the risk of a discontinued support was rated to be low. The most possible risk that the work is facing with the decision to use WandB is a temporary unavailability of the service or a quota exceeding. Beside this risks, the advantages of using a modern experiment tracking service prevail and is often recommended in the data science community.

3.6.2 Augmentation

As second part of the experiments of this projects, the selected augmentation techniques are added to the baseline pipeline. This section aims to give detailed information about the structure, execution and results logged by this experiments, such that it allows to reproduce or extend the experiments in future works as well as explaining the decision path made during the implementation.

Structure

Generally, the augmentation experiments are structured into the same three levels as the baseline experiments. Namely from top to bottom, experiment level, trials level and run level. Though, the implementation of those levels differs from the one introduced for the baseline experiments.

For all selected augmentation techniques, an augmentation experiment encompasses a set of trials. However, experiments are not only conducted for a distinct augmentation technique, but also for each data set, the three selected models and the two training scenarios. So basically, an experiment E_{aug} is run for each unique combination of augmentation technique A, data set D, model M and training scenario S in a grid-search approach.
$E_{aug} = (a \in A,\, d \in D,\, m \in M,\, s \in S)$

In each experiment, a specific augmentation technique is evaluated. For this particular technique, a single primary hyperparameter θ_A is tuned to limit the number of iterations. Therefore, the number of trials within an experiment depends on the distinct values selected for the primary hyperparameter of an augmentation technique (Number of trials $= |\theta_A|$).

For the last level, the runs, the same idea is used as in the baseline experiments. This means that depending on training scenario, the specific cross-validation is performed. The resulting structure of the augmentation experiments is visualized in figure 3.5

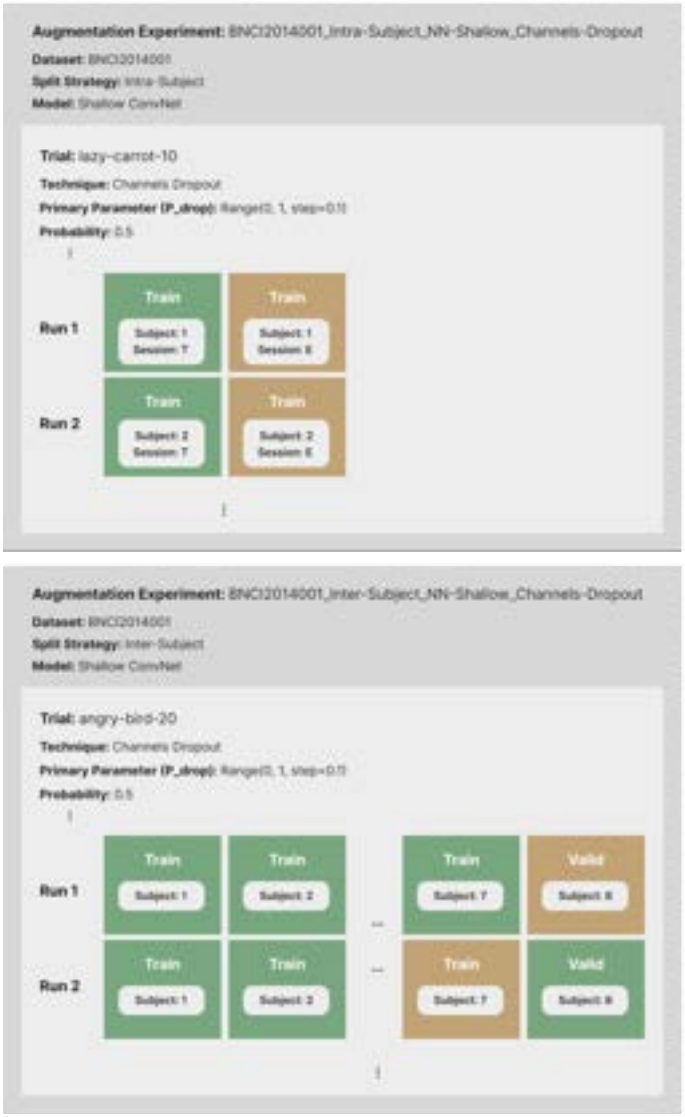


Figure 3.5: Augmentation Experiment Structure

Execution

To limit the number of trials executed within an experiment, only a single hyperparameter is evaluated for each augmentation technique. Since some of the methods have multiple degrees of freedom, the primary hyperparameter to be tuned must be determined.

All techniques consists of a probability p_{aug} with which a certain technique is applied to a sample. Because p_{aug} is not a method specific parameter and rather describes by which amount the original training set distribution should be extended, it is not considered as parameter of interest. Thus, p_{aug} is set to 0.5 for all augmentation experiments of this work.

The remaining hyperparameter for Channels Dropout is p_{drop}, for FT Surrogate the phase noise magnitude and for Smooth Time Mask the length and the position of a mask within a sample. In this work, the length of the mask is used as the hyperparameter to be evaluated and the position is chosen randomly.

To be able to observe the sensitivity of a technique to its evaluated hyperparameter, ten trials are conducted for each method. Both Channels Dropout and FT Surrogate have a bounded hyperparameter space by an interval between [0, 1]. Therefore, a step of 0.1 is applied starting by 0.1. For the Smooth

Time Mask, a maximum length of two seconds is considered appropriate, as the length of the experimental window is four seconds for the BNCI2014001 data set and 4.5 seconds for the BNCI2014004 data set. Starting with a length of 0.2 seconds and increasing in steps of 0.2, the maximum length of the zero mask makes up about 50% of the input sample. A summary of the tuned hyperparameter for each augmentation technique can be found in table 3.3.

Method	**Tuned Hyperparameter**	**Other Hyperparameters**
FT Surrogate	Phase Noise Magnitude *Range(0, 1, step=0.1)*	P_{aug}: 0.5
Channels Dropout	Dropout Probability *Range(0, 1, step=0.1)*	P_{aug}: 0.5
Smooth Time Mask	Mask length (s) *Range(0, 2, step=0.2)*	P_{aug}: 0.5 Mask position: picked randomly

Table 3.3: Tuned Hyperparameter Space Augmentation Experiments

Results

The intention of the result visualisations is to show the relative change in accuracy when applying a defined augmentation technique with a certain hyperparameter value. In more detail, for each selected augmentation strategy a plot should be generated which displays the relative change in accuracy on the Y-axis and the values of the hyperparameter on the X-axis. Because in the intra-subject setting the augmentation technique gets validated for each subject, a box plot can be displayed to see how much variance exists between different subjects. A box plot was chosen instead of error bars to get a sense of the true underlying distribution of the result data points. Moreover, error bars rely on a Gaussian distribution assumption, which can not be satisfied with the number of subjects offered by the used data sets. Similar, a box plot can be displayed for the inter-subject training scenario due to the use of a LOOCV over the training subjects. An example of a result plot for the Channels Dropout augmentation technique is shown in figure 3.6

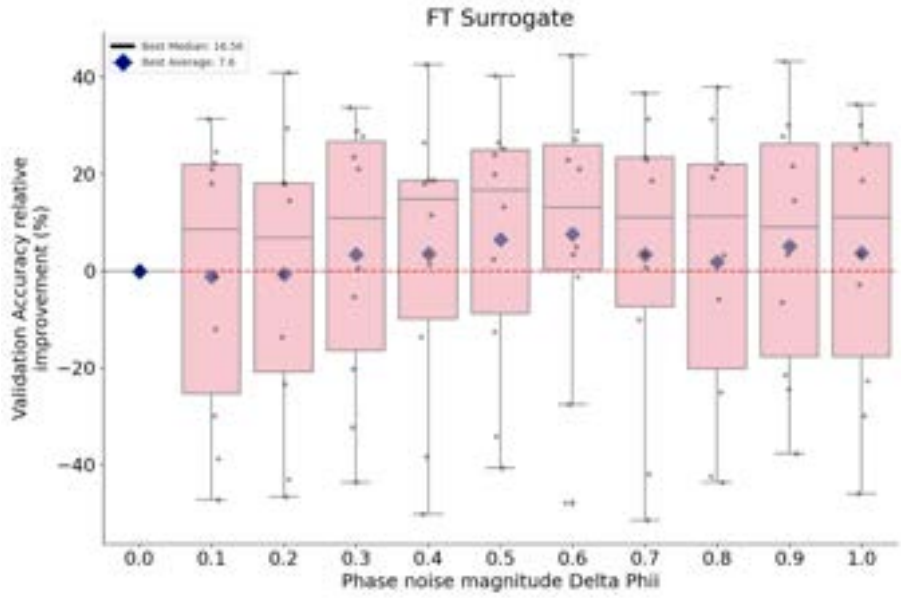


Figure 3.6: Channels Dropout Result Example

Chapter 4

Implementation

A contribution of this project is to build a data augmentation pipeline that can be flexibly applied to different models, augmentation methods or data sets. The focus of this chapter is to describe the technical details about the pipeline setup and should indicate what kind of adjustments to the pipeline can be made to reproduce or even extend the experiments.

4.1 Libraries

First of all, Braindecode was used to implement the preprocessing steps, integrate the augmentation strategies as well as defining the two CNNs of this project [36]. The implementation of the neural models in Braindecode is done via Pytorch together with a Skortch wrapper. This allows the usage of all layers, callbacks, data loaders and criterions provided by Pytorch [121;122]. The Skorch wrapper enables to use Scikit-Learn methods along Pytorch and offers an aligned API [122;80]. Scikit-Learn itself was used to implement the classical machine learning models [80].

4.2 Requirements

At the time of the project setup, `Python` version `3.9.x` was required to be compatible with the dependencies used in the project. The package preventing the project from using `Python 3.10.x` was `skortch` [122]. However, `Python 3.10.x` support of the package was under active development by the time of the project. All required dependencies can be found in the `requirements.txt` in the root folder of the project. Additional information about the environment setup is given in the `README.md` file.

After running the pipelines, all resulting plots are stored in the project's `results` folder. Beyond the visual plots, the project depends on the platform Weights & Biases (WandB) for metrics and progress logging [120]. However, it is not mandatory to use WandB to run the project and can rather be configured in a dedicated configuration file. For security reasons, the credentials to connect to the WandB project are stored in a separate `.env` file, which is not checked in to the version control system. The required environment variables to successfully connect to WandB are also explained in the `README.md` file.

4.3 Setup and Structure

The project is structured into four scripts, which are stored in the `scripts` folder of the project. First, the `main.py` script which executes the baseline and the augmentation experiments. The script is structured into different steps such as loading and preprocessing the data set, training and evaluating the model or logging and storing the results. For each step different configurations can be applied which are presented in section 4.5. Second, the `dataset_exploration.py` script which can be used to get a first impression of a dedicated data set. On the one hand, it visualizes an example of each class within a data set. On the other hand, it plots the distribution of the classes in a histogram to validate if classes of a data set are balanced. The third script called `augmentation_visualization.py` can be used to visualise the effects of the introduced augmentation techniques. Lastly, the `evaluate_experiments.py` script validates the results obtained during the baseline and augmentation experiments by plotting the desired graphs on the basis of the queried data from WandB.

All scripts allow a number of configurations to simplify the execution of different experiments and to adapt certain diagrams to the respective needs. The configurations are stored in a tree of `YAML` files in the `config` folder of the project. In the upcoming sections, the level of possible configuration is presented as well as the effort required to extend the experiments and its visualisations.

4.4 Experiment Pipeline

The experiment pipeline, implemented in `main.py`, is divided into different steps and visualized in figure 4.1. A typical machine learning pipeline with data loading, preprocessing, training and evaluating was set up. In addition to that, a second preprocessing step to create several windows from a single trial is applied to increase data efficiency. Also the model needs to be initialised first to evaluate the model its receptive field size, which is required to decide about the window stride during the window cropping.

With the received window data set, the defined split strategies can be applied to divide the data set into a training, validation and test set. The classifier initialisation is only required for neural based models of the pipeline. The instantiated Skorch classifier handles the training of the Pytorch based model such that it can be used like an ordinary Scikit-Learn classifier. Moreover, training parameters such as batch size or the number of epochs as well as the optimizer is configured in this step of the pipeline.

Before training, the augmentation techniques get initialised. While for the neural track the augmentation methods are applied online during training, it is required for the basic track to apply them directly to the training set beforehand.

After training, the model is used to predict the classes on the test set and the defined performance metrics are computed as well as the plots generated. In the last step all metrics, plots and the meta data created are stored on the WandB platform.

Two subsequent iterators around the pipeline are required to conduct the desired experiments. One to tune a model's hyperparameters in the baseline experiments or to validate an augmentation technique's primary hyperparameter during the augmentation experiments. A second is used to implement a cross-validation-like structure depending on the training scenarios.

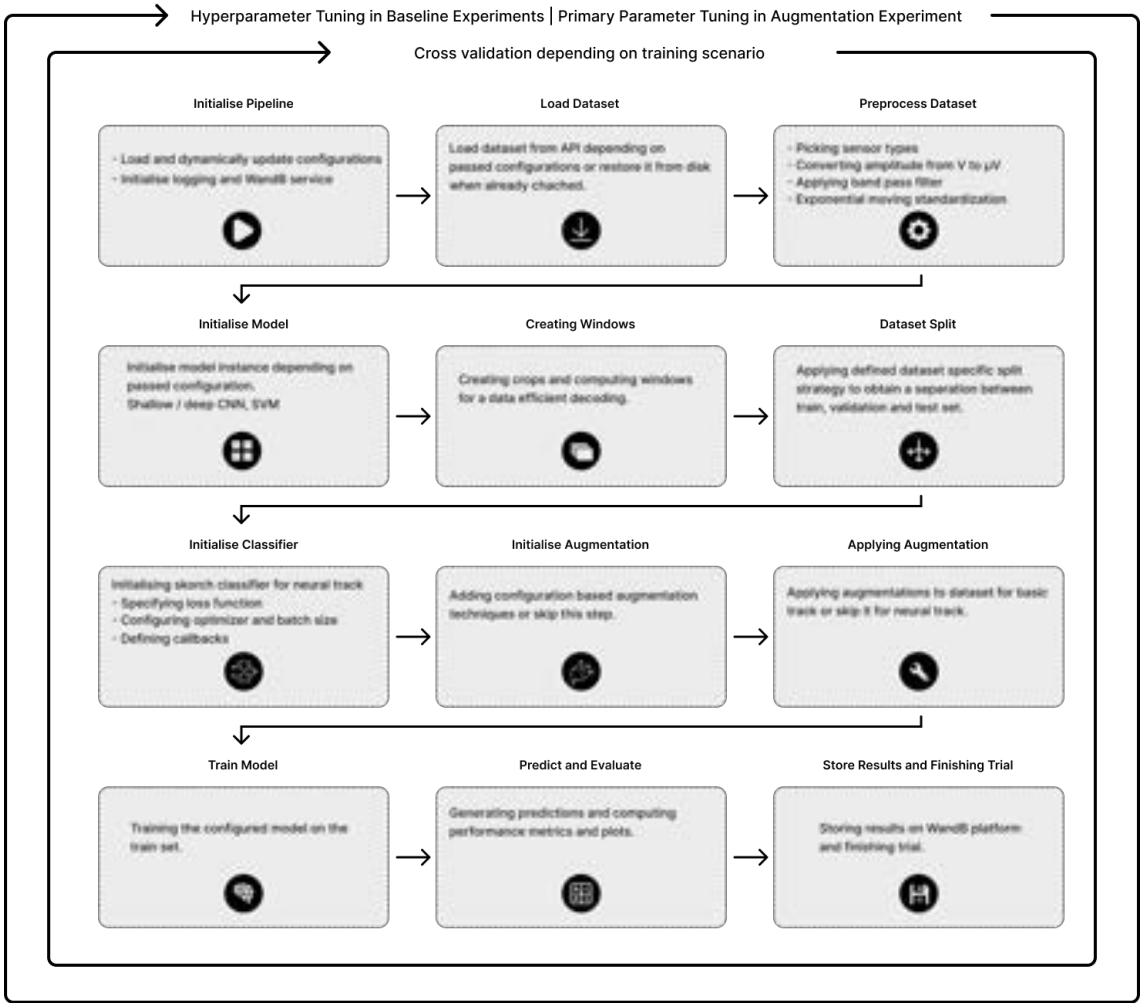


Figure 4.1: Experiment Pipeline

4.5 Experiment Configuration

For both, the baseline and the augmentation experiments, the implemented pipeline allows a defined set of configurations. This section aims to give a basic understanding on this configurations and how they can be used to customise the experiments according to the needs. The `config.yaml` file within the `config` folder is used as entry point to all experiments related configurations. All other configuration files are then resolved in a tree like structure by the Hydra plug-in, such that eventually a configuration dictionary can be access within the invoked function [118].

Generally, the pipeline supports all data sets offered by the MOABB API [116]. However, to use a particular data set within the pipeline, some information is required, which is stored in the data set configurations files in the `/config/dataset` folder. Currently, the project supports six motor imagery data sets. Depending on the data set, a different split strategy is implemented. If no customised split strategy to a certain data set is defined, the pipeline defaults to a fractional split of the subjects. Additionally, it can be configured if an intra- or inter-subject training scenario should be applied for the data set. To run the pipeline with a particular data set, the `dataset` key in the `defaults` section of the `config.yaml` file can be set according to the data set name (referencing the YAML filename in the `/config/dataset` folder).

During the preprocessing step of the pipeline, the project uses the proposed configurations by the Braindecode library [36]. Despite, parameters like the range of the band pass filter or adjusting the offset

of trial window can be set in the preprocessing configurations file. Compared to the examples in the Braindecode library, the number of parameters required was reduced when it was possible to calculate them using the information provided in the data set.

The pipeline supports four configurable models. A shallow and a deep CNN, a Support Vector Machine and a Random Forest. Each model obtained a separate configuration file in the `/config/model` folder. The main idea of the model configuration files is to set a defined number of hyperparameters for the model. Moreover the pipeline differentiates between neural network based and classical machine learning models. For neural based models, additional parameters for training and optimizing the model such as batch size or learning rate can be set disparate configuration files (`/config/training` and `/config/optimizer`).

In order to define the hyperparameter search space in a baseline experiment, a Optuna sweeper is used [119]. With Optuna, for each hyperparameter a range or a choice of values can be defined. If this two methods are not sufficient, also a custom strategy can be implemented. Additionally, a maximum number of experiments and an optimizing direction of the returned metric (minimize / maximize) needs to be set. The hyperparameter search space of the project can be set in the `hydra/sweeper/params` section of the `config.yaml` file. To run a baseline experiment with a hydra multi run configuration, the `run_baseline_experiment.sh` shell script can be executed.

Six augmentation techniques are implemented in the pipeline and can be switched on in the `config/augmentation` configuration file. For each unique method, the values of all supported hyperparameters can be set. In addition, the primary hyperparameter that is optimised during the augmentation experiments can be configured together with its search space. Augmentation experiments can be run by executing the `run_augmentation_experiment.sh` script.

The project supports two types of result visualizations. Either the results can be plotted as a box plots or in favour of comparability to the work of Rommel et al. [27] they can be visualized as point plots extended with error bars. Configurations to the plots such as the referring baseline experiment or technique depended colours or labels can be set in the `config/visualisation` file. To create the result diagrams, the name of the augmentation experiment must be set in the configuration file, followed by the execution of the `evaluate_experiments.py` script. This because the visualisations depend on the data stored on the WandB platform, which is identified by the name of the experiment.

Lastly, the configuration of the WandB integration can be adjusted. The files in `config/wandb` represent the logging of two environments, one for development and one for conducting the experiments. This separation is intended to prevent experimental and test data from being mixed on the WandB platform. Furthermore, the WandB platform supports different messaging channels to inform about the state of the running experiment. Therefore, if set up accordingly on the platform, messaging can be toggled from within the configuration file.

4.6 Experiment Extension

If the configurations of the pipeline are not sufficient for the needs of the experiment, the pipeline can to a certain extend be extended effortlessly. For each step of the pipeline there is a separate service with atomic functions that are switched by the given configurations. This means that the most of the components used by the pipeline can be extended by creating a new component and making that available through a dedicated configuration file.

First, the pipeline can be expanded to support all the data sets available via the MOABB API. In order to add a data set, only an additional configuration file in the `/config/dataset` folder needs to be created according to the existing templates. If also a dedicated split strategy needs to implemented, a named function can be added to the `DatasetService` class. Moreover, many utility functions to split a data set by its hierarchical entities exist, which can be combined sequentially to get to the desired result.

Because the pipeline is based on Pytorch [121], Scikit-Learn [80] and the Pytorch wrapper Skorch [122], custom models based on both libraries are supported by the pipeline. A new model can be added by

extending the `ModelBuilder` class by a new model and adding it's hyperparameter configurations to a new file in the `/config/model` folder.

Likewise to the models, the evaluated metrics can be extended effortlessly by adding a new method to the `WandBService` class. Since the index and probability predictions of the classes for all data set partitions are passed to the service, new metrics offered by different libraries can be plugged in easily. Examples of metrics exists from Scikit-Learn [80], Pytorch [121] and Torchmetrics [123].

Finally and most importantly for the project, the pipeline can be extended to support all augmentation techniques offered by the Braindecode library. In order to add a certain method, a new atomic generator function, which instantiates the augmentation class, needs to be created in the `AugmentationService` class. Similar to the existing examples, a new configuration file for the included technique can be added to the `config/augmentation/augmentation.yaml` file. Custom augmentation techniques can be implemented by extending the `Transform` class of the Braindecode library, which is basically a `Pytorch Module`.

To sum up, the modular structure and the use of multiple prevalent libraries allows to extend the pipeline extensively. Depending on the degree of customization required, a different abstraction level of the libraries can be used to implement the desired strategies.

Chapter 5

Results

Numerous experiments were conducted according to the experiment plan attached in appendix A.2.2. The aim of this chapter is to summarise the results of the baseline and then those of the augmentation experiments.

5.1 Baseline Experiments Results

The result visualisation of the baseline experiments is divided into the two main applied train scenarios: intra- and inter-subject. For each scenario, the best result, based on the average of the cross-validated runs, is presented for all models and data sets. In addition to the number-based metrics, the created performance plots are visualised exemplarily.

5.1.1 Intra-Subject

With the four classes of the BNCI2014001 data set, the classification task is significantly more difficult for the model than with the two classes of the BNCI2014004 data set.

In order to assess the performance of the models comprehensively, the average of all performance metrics were calculated for each trial. For the intra-subject scenario this means the average over the subjects of a data set. The results with the highest average achieved by the different models are listed in Table 5.1.

Surprisingly, even with the proposed hyperparameters of Schirrmeister et al. [36], the deep CNN was not able to exceed the performance of the shallow CNN. Rather more predictably, the SVM was not able to significantly surpass the random chance of 25 % for the four class data set and the 50 % boundary on the two class data set. Because of the small number of subjects in the data sets ($|subjects| < 11$), even with the use of the *t-distribution*, a confidence interval cannot be used for an accurate conclusion. Therefore, the minimum and the maximum value of the test accuracy are listed in the performance table in order to assess the variance between subjects. By consulting the table, an expected high inter-subject variability can be observed. Despite the lower average performance achieved by the deep CNN, it shows a slightly smaller variance in its results. The variance achieved by the SVM is questionable and origins more likely on the marginal performance on all subjects.

A second characteristic of the results is the high amount of overfitting when using neural based approaches. The fact that even the shallow model is able to fit the test data perfectly contradicts slightly the assumption of EEG data as very complex high dimensional data. However, the high bias value also implies that with additional data of the true underlying distribution, the variance could further be reduced. Also, the high accuracy attained for certain subjects, in combination with the observation of

the converging learning curves, it can be concluded that the model is able to learn how to discriminate between classes for this type of data. A sample learning curve can be seen in appendix A.2.4.

A first impression about the results could be acquired by considering the number-based performance metrics. In order to get more in-depth insights, three representative subjects achieved by the shallow CNN on the BNCI2014001 data set are visualized in figure 5.1. Additionally, the results obtained on the BNCI2014004 data set can be viewed in figure A.3 of the appendix. The figures show that the performance of the models highly depends on the different subject. Also the classes on which the model has more difficulty or shows the most confusion vary between subjects. This means that for some subjects there exists most confusion between left and right hand and for others among right hand and feet. Furthermore, these characteristics of the results can be determined independently of the underlying data set or the model used. Therefore, the results highlight the problem of high inter-subject variability when using EEG data for classification tasks as described in the literature.

Model	Median Acc.	Avg. Acc.	Min. - Max. Test Acc.	Test F1-Score
Shallow CNN	Test: 72.92 % Valid: 79.69 % Train: 100.00 %	Test: 68.81 % Valid: 71.99 % Train: 100.00 %	[40.10, 88.02] %	67.95 %
Deep CNN	Test: 67.36 % Valid: 73.96 % Train: 100.00 %	Test: 57.94 % Valid: 60.16 % Train: 99.91 %	[41.67, 77.08] %	57.05 %
SVM	Test: 28.13 % Valid: 26.82 % Train: 25.52 %	Test: 26.91 % Valid: 26.42 % Train: 25.21 %	[18.75, 30.73] %	25.26 %

Table 5.1: BNCI2014001 Baseline Intra-Subject Experiments Results

Model	Median Acc.	Avg. Acc.	Min. - Max. Test Acc.	Test F1-Score
Shallow CNN	Test: 83.75 % Valid: 83.13 % Train: 91.00 %	Test: 79.10 % Valid: 79.17 % Train: 89.29 %	[51.56, 94.37] %	78.92 %
Deep CNN	Test: 84.38 % Valid: 86.25 % Train: 100.00 %	Test: 79.44 % Valid: 81.39 % Train: 100.00 %	[48.75, 94.69] %	79.32 %
SVM	Test: 53.13 % Valid: 50.94 % Train: 36.88 %	Test: 53.40 % Valid: 51.55 % Train: 43.26 %	[50.94, 55.94] %	52.62 %

Table 5.2: BNCI2014004 Baseline Intra-Subject Experiments Results

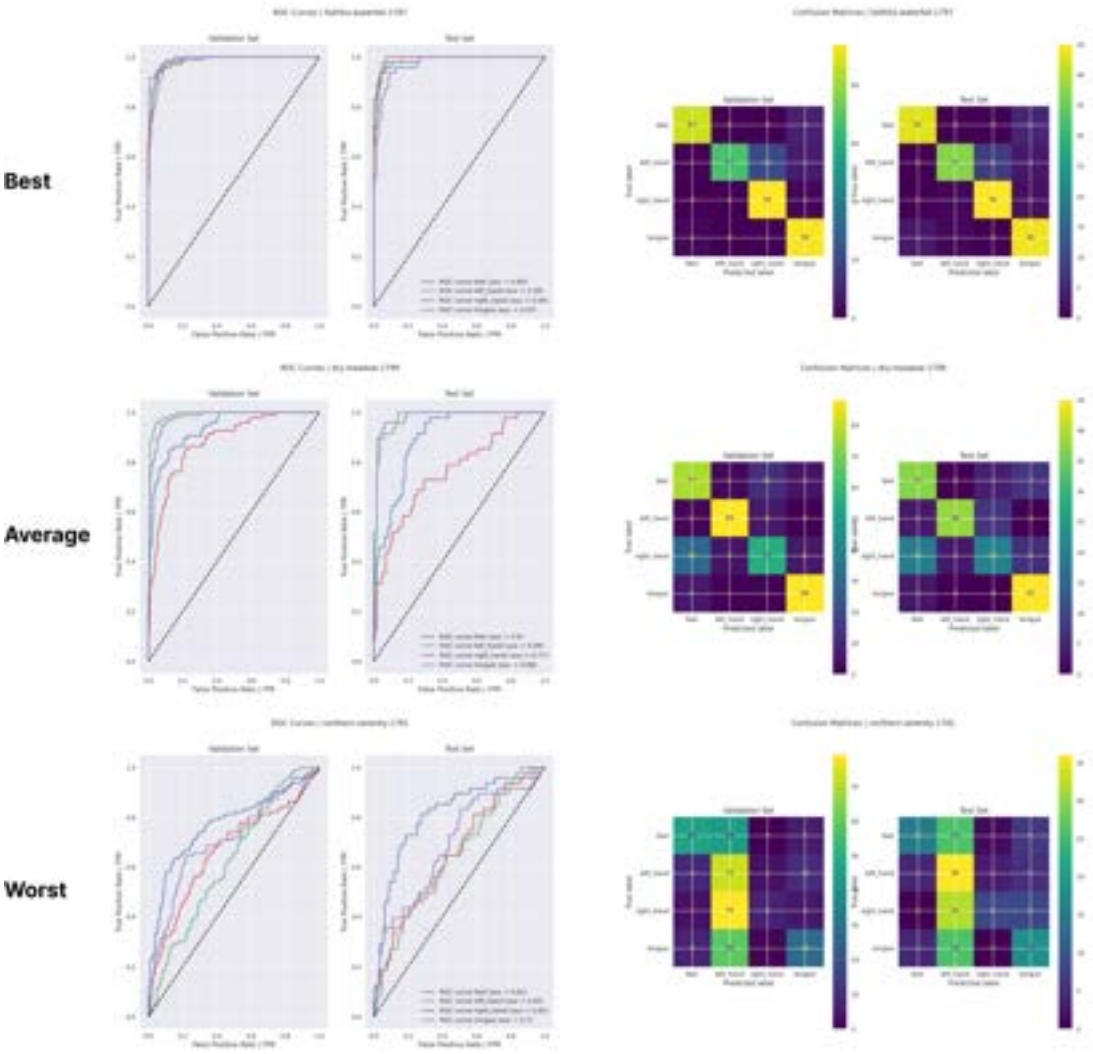


Figure 5.1: Intra-Subject Baseline Experiment Performance Evaluation Shallow CNN on BNCI2014001

5.1.2 Inter-Subject

Similar to the intra-subject experiments, it is the aim of this section to present the results of the inter-subject experiments likewise, which are listed in table 5.3. In contrast to the intra-subject experiments, the average performance per trail was calculated not based on the individual subjects, rather by using the training subjects with a LOOCV strategy. Alike, minimum and maximum performance values are listed based on the different LOOCV runs of a trial and not by the unique subjects as in the intra-subject scenario.

As the amount of data and difficulty of the task increased, the average performance for the BNCI2014001 data set decreased significantly. However, it is important to note that due to the limited number of subjects in the BNCI2014001 data set, only one static subject is used as test data set. To reduce the decline in performance to this fact alone would not be sufficient, since not only the test accuracy but also the cross-validated validation accuracy decreased, although not as drastically. Furthermore, a first difference between the data sets can be observed. The drop in performance did not occur under the application of the BNCI2014004 data set and could rather be improved. Here the reason is clearly referable to a beneficial subject used as test set. A may more realistic value can be considered with the cross-validated validation accuracy. Nevertheless, for the BNCI201004 data set, the performance is almost at the same level as in the intra-subject training scenario.

A further notable difference to the intra-subject setting can be observed by considering the overfitting capabilities of the models. While the deep CNN still overfits strongly on the training set, this is no longer the case with the shallow CNN and the larger amount of data. Especially for the BNCI2014004 data set, the bias error decreased significantly. But still, despite the ability to overfit, the shallow CNN still achieves a better validation and test accuracy for both data sets.

Interestingly, as the amount of data increased, the SVM was able to produce better results than in the intra-subject scenario. Also, the balanced data sets do not lead to a notable difference between the test accuracy and the weighted F1-Score. Finally, the difference between the minimum and maximum values of the runs has decreased considerably. This shows that the test accuracy is not expected to change drastically when removing a specific subject from the training set unless some of the remaining subjects are still valuable.

Beside the effect on classification performance, the increase in training data has a strong influence on the running time of the training. While in an intra-subject scenario a run took about one minute for the shallow CNN, 1.5 minutes for the deep CNN and 15 seconds for the SVM, the required time increased to six minutes for the shallow CNN, 15 minutes for the deep CNN and three hours for the SVM. Because the inability of the SVM to process the data within a time amount respecting the scope of this project, the SVM is only evaluated further for the BNCI2014001 data set when using an inter-subject scenario. Approaches to tackle this shortcomings of the SVM by applying additional preprocessing to the input are discussed in chapter 6.

By the comparison of different runs of the BNCI2014001 data set and the application of the shallow CNN in figure 5.2 more details can be explored. Additional examples on the BNCI2014004 data set can be found in the appendix in figure A.4. In the LOOCV between subjects as a validation sets, again a high intra-subject variability is observed. When considering the validation ROC curve of the run with the highest accuracy on the BNCI2014001 data set, the model could merely discriminate between the different classes, whereas it in other runs achieved a reasonable value for the AUC metric. Since the same subject was used as test set, it is evident that the model has difficulties with the same classes over all runs, namely feet and tongue. Throughout testing, the model clearly overrated the right hand class even with a balanced data set. This is a clear evidence, that the model is not capable to distinguish between classes based on the appropriate characteristics of the features for this particular subject. However, this outcomes gets contrasted by the results on the BNCI2014004 data set where the model achieved similar AUC values for both classes. Also, the worst run shows that multiple valuable subjects need to be present in training set to achieve an accuracy level that is significantly better than random chance.

Model	Median Acc.	Avg. Acc.	Min. - Max. Test Acc.	Test F1-Score
Shallow CNN	Test: 37.85 % Valid: 35.42 % Train: 69.70 %	Test: 38.32 % Valid: 42.96 % Train: 69.70 %	[35.85, 41.32] %	35.08 %
Deep CNN	Test: 37.50 % Valid: 40.80 % Train: 100.00 %	Test: 37.71 % Valid: 46.18 % Train: 99.97 %	[34.84, 41.90] %	34.46 %
SVM	Test: 31.60 % Valid: 30.90 % Train: 99.65 %	Test: 31.55 % Valid: 30.90 % Train: 99.63 %	[30.47, 33.07] %	28.13 %

Table 5.3: BNCI2014001 Baseline Inter-Subject Experiments Results

Model	Median Acc.	Avg. Acc.	Min. - Max. Test Acc.	Test F1-Score
Shallow CNN	Test: 90.41 % Valid: 74.46 % Train: 74.72 %	Test: 90.30 % Valid: 70.24 % Train: 75.33 %	[88.72, 91.49] %	90.25 %
Deep CNN	Test: 86.69 % Valid: 71.32 % Train: 97.97 %	Test: 87.10 % Valid: 67.61 % Train: 95.97 %	[84.19, 89.76] %	91.15 %

Table 5.4: BNCI2014004 Baseline Inter-Subject Experiments Results

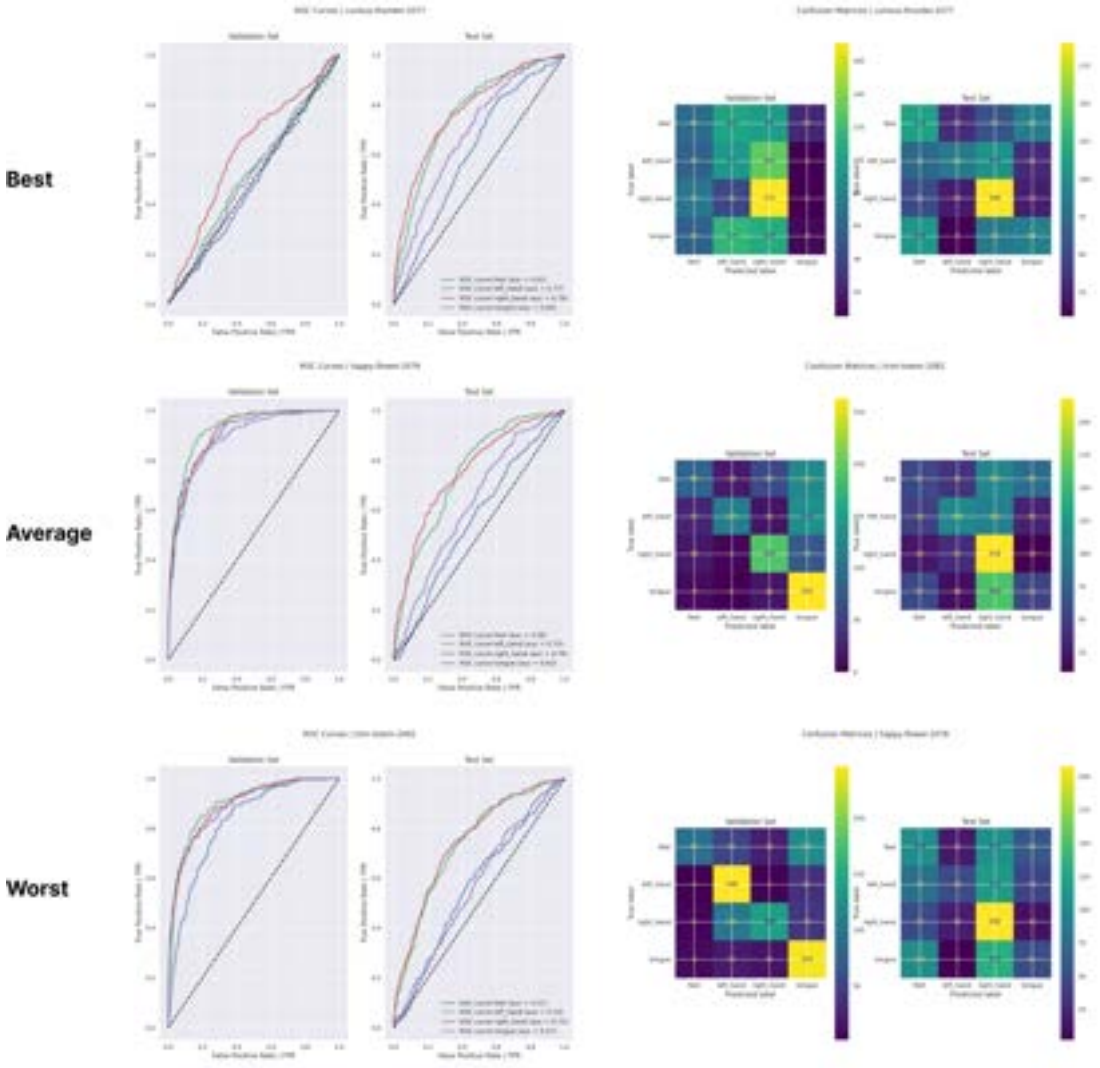


Figure 5.2: Inter-Subject Baseline Experiment Performance Evaluation Shallow CNN on BNCI2014001

5.2 Augmentation Experiments

The results of the augmentation experiments are divided into the three different investigated augmentation strategies. For each strategy, the results are organised by the applied training scenario. Then for both scenarios, a grid with the resulted graphs is displayed, in which the X-Axis describes the models used and the Y-Axis distinguishes between the data sets.

5.2.1 Channels Dropout

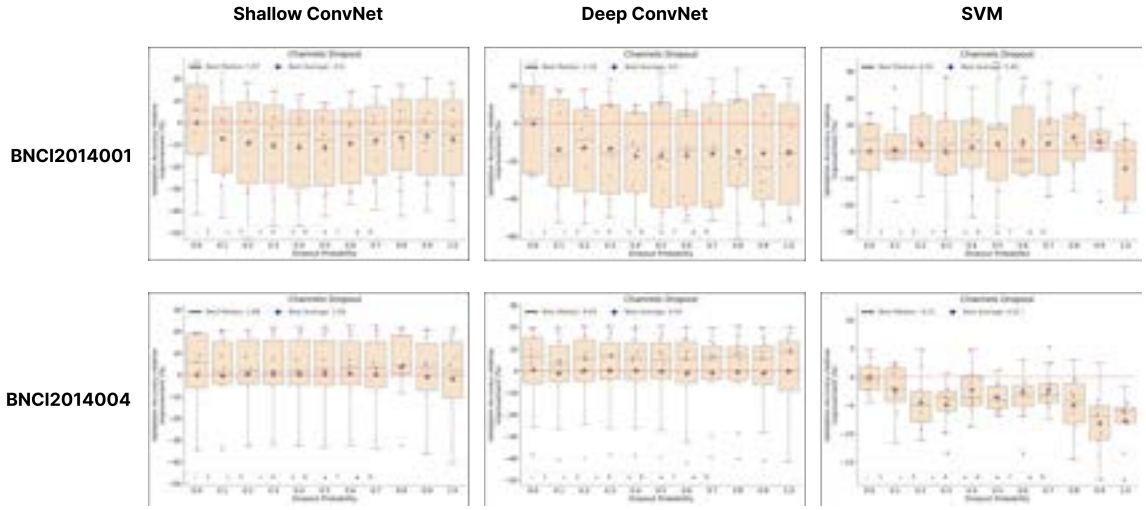
As first augmentation technique, Channels Dropout was analysed. The resulting two grids for the intra and inter-subject training scenario are visualized in figure 5.3.

For both training scenarios, the average and the median performance did not increase significantly, regardless of the data set. The best effect of Channels Dropout can be observed for the BNCI2014004 data set in an intra-subject scenario, where a tendency to reduce variance exists for the neural-based models. In addition, the model is more sensitive to hyperparameter changes of p_{drop} for larger data sets in an inter-subject scenario, as only small variations of accuracy can be detected for the intra-subject scenarios.

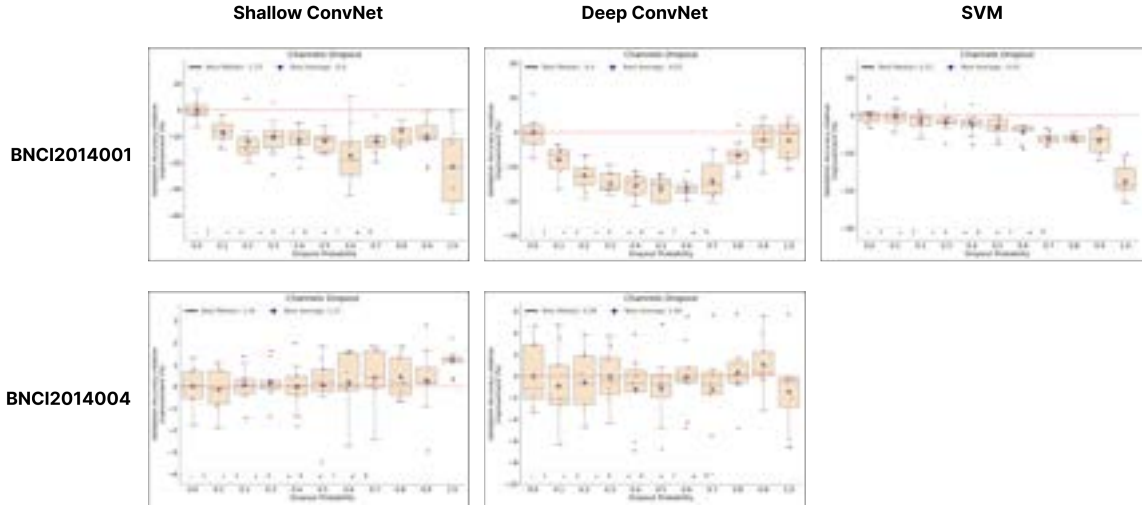
The inter-subject variability can clearly be observed for the intra-subject experiments. In general, the technique cannot drastically change the performance order of the subjects. This means that if the subjects are divided into three clusters, it is not likely that the application of Channels Dropout is changing the members of this clusters. However, for the deep CNN on the BNCI2014004 data set, one can see that there are subjects like number nine that benefit more from CD than others. There is also a tendency to identify some difficult subjects, such as number two, who lead to poorer results across models when using CD.

By comparing the results on the WandB platform, applying the Channels Dropout method reduced the variance by simultaneously also decreasing the general classification performance in most cases. Also, a carried out variance analysis shows a high negative Spearman and Pearson correlation of a reducing variance when increasing p_{drop}. An example for such a variance analysis for Channels Dropout applied on the BCNI2014004 data set can be found in appendix A.2.5. The reason why the regularization effect of Channels Dropout is not directly visible is that it can only be observed for subjects on which the model discriminated reasonably also without applying the technique. For challenging subjects, the use of Channels Dropout tends to lead to a deterioration of the results. Eventually, even if Channels Dropout acts like a regularization technique, the amount of overfitting remains considerably high.

To sum up, Channels Dropout has a larger effect on more extensive data sets. A model can benefit from the method by making the model less dependent for specific subjects. Nevertheless, its effect depends on the prevalent data set. This also leads to the conclusion that under the applied split strategies, the results reported by Rommel et al. could not be reproduced for the Channels Dropout technique and transferred to other data sets [27].



(a) Results Channels Dropout | Intra Subject



(b) Results Channels Dropout | Inter-Subject

Figure 5.3: Results Channels Dropout

5.2.2 FT Surrogate

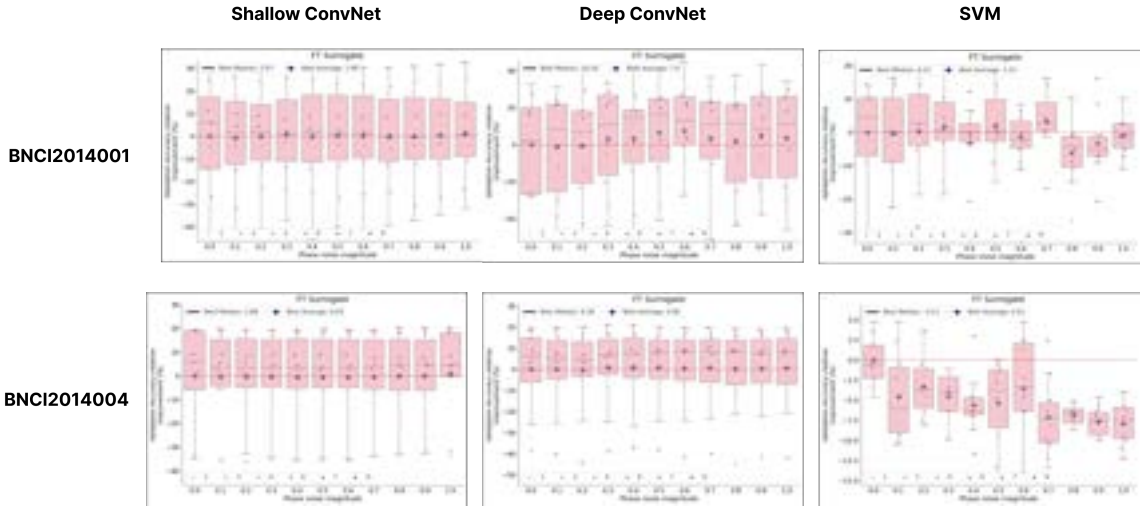
The application of FT Surrogate shows a different results in comparison to Channels Dropout. As shown in figure 5.4, FT Surrogate has a positive effect in most of the intra-subject experiments. By using the deep CNN in an intra-subject scenario, the highest accuracy increase of 7.6% could be achieved.

If the subjects are clustered according to the level of relative accuracy increase, the higher the value for phase noise magnitude gets, the lower the variance within this clusters becomes. This means that the between subject variance could not be narrowed over the entire population of subjects, but for those in a certain performance range. Consequently, the average does not change significantly and the median can even drop slightly. However, it could also lead to a more consistent result in a specific setting.

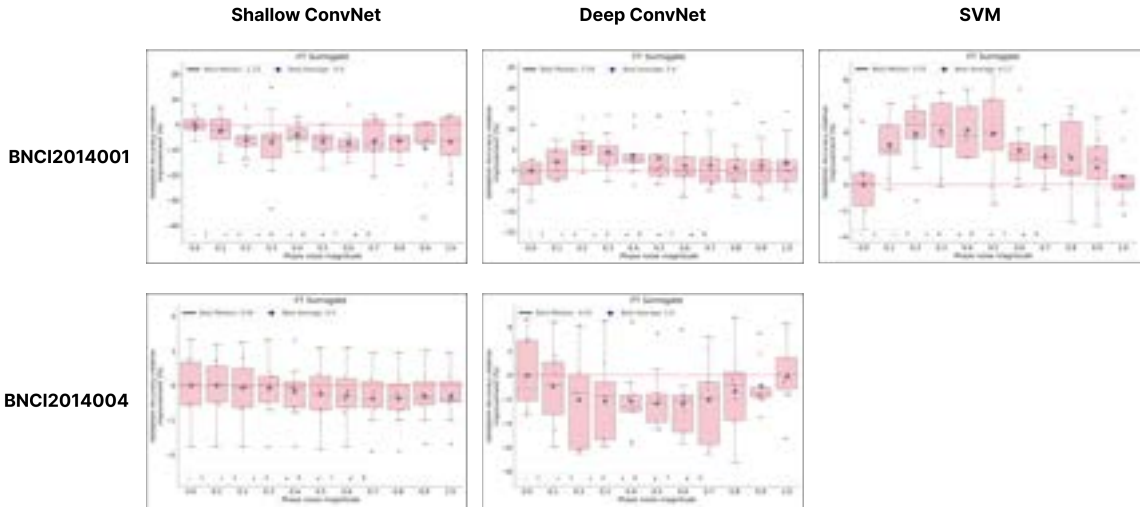
A different result shows the application of FT Surrogate in an inter-subject scenario. Only using the deep CNN on the BNCI2014001 data set resulted in a positive change in accuracy. There is also no significant positive effect on the variance perceptible if exchanging one subject during training. However, challenging or well discriminable subjects can clearly be identified for both neural based models, which could allow further analyses on what makes them difficult.

When considering the results to assess overfitting, an akin situation to the Channels Dropout technique can be observed. Classification error increases while variance decreases slightly. Also when assessing the conducted variance analysis, there is a clear negative correlation between variance and an increasing phase noise magnitude value. But even if the generalisation error drops and the method could be applied as a regularisation technique, there is often a shift in performance on both data sets, making the positive overfitting reduction redundant.

To conclude, FT Surrogate also does not show a general trend that its use is likely to have a positive effect when using a particular type of model or regardless of the data set.



(a) Results FT Surrogate | Intra-Subject



(b) Results FT Surrogate | Inter-Subject

Figure 5.4: Results FT Surrogate

5.2.3 Smooth Time Mask

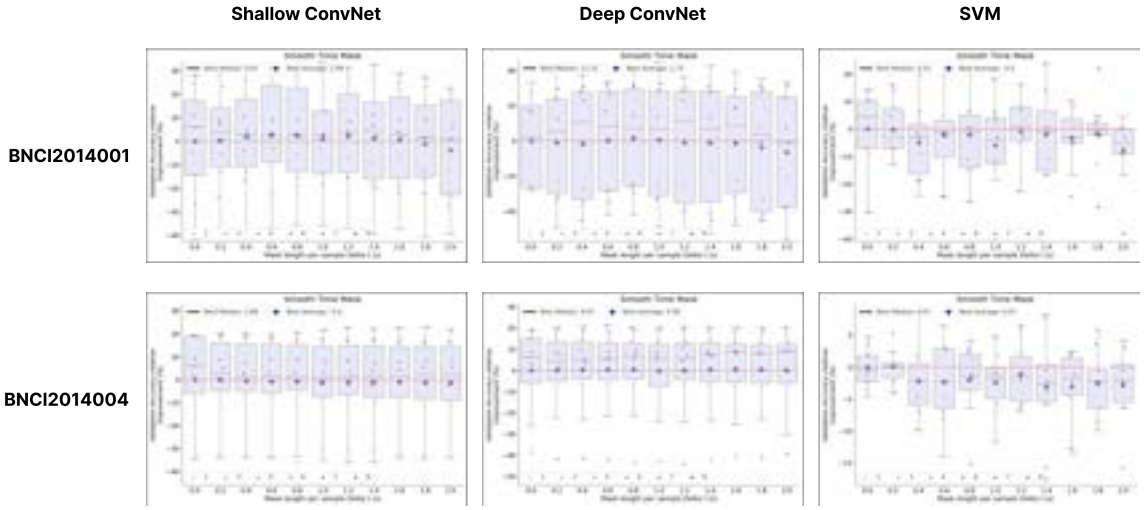
The last augmentation technique considered was the Smooth Time Mask (STM) method. The results obtained are visualised in figure 5.5. In comparison with the other techniques evaluated, it achieves the best overall performance increases. Even if the raise in performance was often marginal, the technique was able to achieve an improvement in most cases. Together with the results of Rommel et al.[111], STM can therefore most likely be seen as suitable augmentation method for motor imagery BCI systems based on EEG data, which performs for different models and irrespective the data set. Moreover, the highest relative accuracy increase of 8.56% could be achieved when using STM.

If the training scenarios and data sets are considered in more detail, the results show only a moderate sensitivity to the evaluated hyperparameter changes in an intra-subject scenario. A clear evidence that the technique is able to reduce the overall inter-subject variability cannot be observed. There is more likely a tendency that the discrimination for challenging subjects gets even more difficult when using the deep CNN. But even if the average performance could not be improved considerably, the deep CNN shows that it is able to improve the median performance for both data sets. This because of the inter-subject variability reduction between subjects beyond the baseline average. This concludes that STM is more effective on samples with valid information for the model rather than on those which confuse the model. Because STM only removes information from the original signal, this effect is also reasoned.

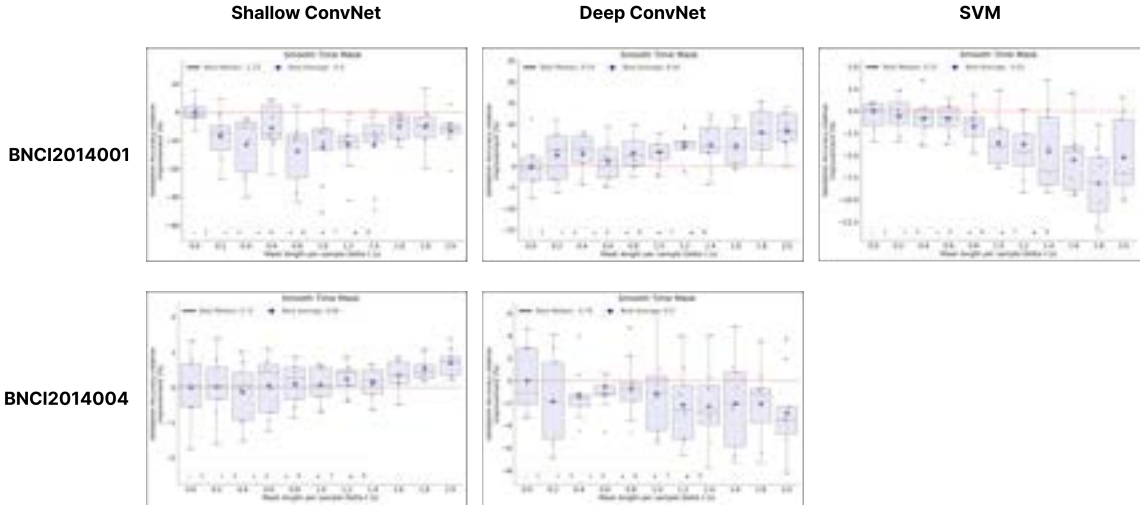
Based on an inter-subject training scenario, using STM is more sensitive to hyperparameter changes. But beside the considerable performance increase on the BNCI2014001 data set, in this scenario, no consistent conclusion can be drawn that the application of the method results in a positive effect independent of the data set or model family used. However, the method can help reduce dependence on a particular subject during training, as the variance for certain hyperparameter value ranges can be reduced on a scenario-specific basis. This can be helpful even if the average performance drops slightly to get a more robust evaluation about the actual performance of the model.

Regarding overfitting, the method shows a slightly different outcome to the methods analysed in the previous sections. Whereas it could reduce overfitting in both training scenario by using the shallow CNN, no reduction of variance could be observed for the deep CNN as the bias error remained almost zero.

To sum up, Smooth Time Mask is a promising augmentation technique for motor imagery EEG data sets. The successful application requires a limited amount of hyperparameter tuning but its transferability to different data sets depends on the applied training scenario.



(a) Results Smooth Time Mask | Intra-Subject



(b) Results Smooth Time Mask | Inter-Subject

Figure 5.5: Results Smooth Time Mask

5.3 Conclusion

The baseline experiments showed that motor imagery classification based on EEG data remains a challenging task even with the high capacity of convolutional neural networks. The most influential aspect on a model's performance is the subject specific data used for training or during evaluation. Independent of the applied training scenario, significant differences could be observed when using specific subjects for training or evaluation of the model performance. Therefore, the conducted baseline experiments highlight, that the high inter-subject variability is one of the major challenge of EEG classification tasks. Additionally, the experiments showed that challenging subjects can be identified regardless the applied models and augmentation strategies.

The high capacity of convolutional neural networks lead to a high overfitting on the training data. Even if EEG data is high dimensional, the sparsity in data, especially in an intra-subject setting, resulted in total overfitting for both neural based models during the baseline experiments.

Between the two data sets investigated, a fairly different baseline performance result was obtained. One reason is the binary versus multi-class setting and another the static, preliminary separated test set. Depending on the selected test subject, performance in an inter-subject setting can vary significantly. However, when it became apparent that the test subject was challenging for the BNCI2014001 data set, it was decided not to replace it and instead to investigate the ability of augmentation techniques to compensate for this subject.

By assessing the baseline performance of the different models evaluated in this project, the results show that both CNN can achieve similar levels of accuracy if the same amount of hyperparameter tuning is done beforehand. In most of the cases, a even slightly better result could be obtained when using the shallow CNN. Opposed to neural based approaches, the use of a SVM without further feature engineering on the input can not successfully be applied on EEG data, as it only marginally exceeded the random classification level defined by $\frac{1}{\#\ of\ classes}$.

Extensive experiments with three models as well as two data sets and training scenarios were conducted to assess the general applicability of three selected augmentation techniques. The effect of the applied augmentation strategies on the average cross validated accuracy by subjects in an intra-subject setting is often marginal with a few exceptions. However, with the deep CNN there is evidence that the median accuracy could be increase in most cases. This shows that the techniques used have a positive effect on subjects where the model is able to reasonably distinguish between classes and a negative effect on those that are challenging. Furthermore, the model was able to reduce the variability between subjects above a certain performance level rather than between all subjects.

In contrast, the techniques applied can have a more sensitive impact on intra-subject experiment results. Both in the positive and negative direction. None of the techniques could be identified to increase the performance irrespective of the data set or independent of the model selected. This leads to the conclusion that with EEG data, augmentation techniques are dependent on the scenario they are used in and a task-specific selection process must be carried out when using the considered techniques in a motor imagery classification task. Another interesting characteristic of all the techniques evaluated is that the results are not very sensitive to a change of the evaluated hyperparameter. It follows that larger steps can be taken in tuning the hyperparameters to find a suitable parameter value for the task to be solved.

Considering the individual methods, Smooth Time Mask followed by FT Surrogate achieved the most promising results regarding classification performance increase. However, Channels Dropout shows the highest tendency of having the ability to make the model less dependent on specific subjects in an inter-subject setting. This effect can be seen in the lower dispersion of results when using the method. Although a significant performance gain of $7-8\%$ could be achieved with Smooth Time Mask or FT Surrogate in a given setting, the high performance gains of Rommel et al. [27] could not be reproduced with the applied training scenarios, even if the same model and the same suggested parameters by Schirrmeister et al. [36] were used for the single setting applied in the paper.

The problem of overfitting with the neural base models could be slightly mitigated with Channels Dropout or FT Surrogate in all the different settings and for both data sets. With Smooth Time Mask

this effect could only be observed with the use of the shallow CNN. Also, even if the variance could be reduced, mostly a shift of the entire performance result can be perceived. In other words, bias error increased more than the gain in test accuracy but still generalization error was reduced moderately.

To sum up, the desired outcome of finding an augmentation technique which can be applied beneficial on different neural based architectures or independent of the motor imagery data set can not be confirmed by the amount of experiments conducted in this project. Nevertheless, the results show that performance gains can be achieved more likely by certain techniques among data sets but with differently tuned models. In addition, the project has shown that difficult subjects can be identified regardless of the used models. The highest relative accuracy increase could be attained by using the Smooth Time Mask method and the deep CNN. There is also a tendency for the deeper CNN to be more sensitive to the application of an augmentation technique than a shallow one. Finally, it was possible to investigate that Channels Dropout and FT Surrogate are able to reduce intra-subject variability within a given performance level and make the model less dependent on specific subjects.

Chapter 6

Discussion and Future Work

The chapter is divided into two sections. First the results are discussed and extensions to the existing solutions are proposed while the reflection. Secondly, possible future work should be prioritised.

6.1 Result Discussion

The novelty of the work was to extensively conduct experiments with specifically selected set of augmentation strategies in different settings of motor imagery classification tasks. Often a dedicated method is tuned to its best in a single setting predefined by a model, a data set and a described training scenario. However, the main focus of this project was to evaluate how general and with what tuning effort a set of techniques can be applied to different motor imagery classification tasks, and to enable conclusions to be drawn about limitations and future work that needs to be done to apply the techniques more effectively.

One of the key findings across all settings tested was that the successful application of an augmentation technique depends on the quality of the input data provided by the subjects of a data set. When considering the results, often only the median value could be increased whereas the average stayed at roughly the same level. A focus on the individual subjects in the point plots shows that the performance on a subject can be improved with the application of an augmentation technique when a certain level of classification performance could be attained for that subject. Additionally, in the inter-subject scenario with a fixed test data set, the performance of the model heavily depends on the subjects selected in the test set. This makes the inter-subject variability also in this work one of the main limitations to achieve better average performance results. A key assumption in machine learning is that the data in the training and test set origin from the same feature space and obey the same probability distribution [124]. This assumption can not be fully met because of the difference between the individual subjects [124]. One method to address this issue is to calibrate the recorded sessions for a long time. However, this is often not feasible due to time and resource constraints and would need a lot of cooperation by the subjects [125;97;126]. Zanini et al. [127] proposed a Riemannian alignment (RA) of the covariance matrices for each trial of a subject to centre them according to the reference covariance matrix in the Riemannian space. Contrary, He and Wu [128] suggested and Euclidean alignment (EA) of the trials which is not dependent on a classifier which operates in the Riemannian space and rather acts on the generated covariance matrices directly. Moreover, EA showed in experiments to be several times faster and only requires unlabelled EEG trials [128;127;124]. Hence, it would be an interesting experiment to apply EA prior to the conducted investigations of this work.

The variability between subjects can also have an effect on the results of the inter-subject experiments. The scenario was implemented taking into account Ng [68] who stated that a test set should be predefined and separated until a conclusive evaluation and should not be investigated in the meanwhile. For this type of EEG experiments this approach could be misleading. Especially if the number of subjects used is small. Still, in a real world application the model should perform approximately equally on all subjects,

but for the experiments of this project, the results in the inter-subject scenario could be misleading with respect to the level of accuracy attained. Therefore, an extension to add a second LOOCV loop that uses each subject as test set independently is proposed. This should give a better insight into the actual result distribution of the subjects.

One of the main advantages of using convolutional neural networks, is that they are able to perform end-to-end learning and do not rely on extensive feature extraction, but learn the features themselves. Therefore only limited preprocessing was applied before training the model. The results showed that a direct comparison between neural based approaches and classical machine learning methods diverges drastically in the performance levels reached. One the one hand, using a SVM classifier directly on the time domain only was able to marginally surpass the random level of guessing. On the other hand, the processing time increased immensely with the increasing data volume in an inter-subject scenario. In recent years, several EEG signal feature extraction methods have been introduced [129]. One of the most popular algorithm family to extract features for motor imagery classification tasks is Common Spatial Patterns (CSP) or one of its variants like Filter Bank Common Spatial Patterns (FBCSP) [129;130;93]. CSP is mainly used to find spatial filters that maximise the variance between the classes of a data set [131]. Although the frequency domain is more important for MI classification, the time domain may contain additional information, especially in a multi-class environment. Therefore, instead of a fast Fourier transform (FFT) or a short-time Fourier transform (STFT), the application of a wavelet transform (WT) would be more suitable for a non-stationary signal such as EEG [132;133]. Beside the manifold of combinations between the mentioned methods, in recent papers, experiments to combine WT with dimensionality reduction algorithms such as Principal Component Analysis or Autoencoders to generate more dense representations of the features were performed [92;134;135]. Finally, for future work, it would be especially insightful to apply a WT based method additionally to the preprocessing steps conducted in the current pipeline.

The claim of Channels Dropout to act as a valuable regularization method can not be entirely confirmed for the experiments of this work. Although there is clear evidence of a correlation between an increasing p_{drop} probability and a decreasing variance, the overall accuracy often could not be raised notably. And also the extent of decrease in variance depends strongly on the subject of investigation. Moreover, in comparison to a state without using CD, there are several subjects on which the variance rose when using only a small p_{drop}. However, while increasing p_{drop} for the same subjects, the variance declined gradually to a lower level than originally without using CD. This results in that Channels Dropout in an intra-subject setting needs to be selected carefully but eventually is able reduce overfitting. Luo et al. [136] propose to use Channels Dropout not by setting a certain channel to zero, but rather using the average of all other channel as the value of the replaced channel. Additionally, Luo et al. [136] suggest not only to apply the technique during training, but also to transform the test set by this method. This leads to additional two strategies to investigate further on the Channels Dropout technique.

The positive result of the deep CNN on the BNCI2014001 data set when using FT Surrogate as augmentation technique shows that at least for some motor imagery data sets, important information lies in the frequency domain. Despite the average on the BNCI2014004 data set could not be improved, the median performance could be raised with both split strategies. Moreover, this proves that FT Surrogate can be applied successfully beyond the original sleep staging task for which it was invented. The fact that the deep CNN is able to benefit more from FT Surrogate could be an indication that later, more high level filter are responsible to extract valuable features in the frequency domain. Strategies to validate this intuition could be visualising the filter outputs in the frequency domain like in the work of Nouri and Azizi [137] or comparing the filter outputs with some handcrafted filters to contrast the outputs between different layers of the CNN as suggested by Tsinalis et al. [138].

The promising outcomes of Smooth Time Mask as a time-domain-based augmentation have shown that the technique is able to reduce the model's dependence on the time domain. The time domain should also not be the primary focus of a model when applied to an MI data set consisting of subjects imagining body movements in a repetitive manner. Hence, time masking could force the model to focus more on the frequency domain and make the model time invariant. This intuition gets backed by the findings of the last paragraph that a deeper model is more able to extract frequency domain specific features and that the shallow network is more dependent on the time domain. Secondly, the hypothesis gets also support by the conclusion that bias error could not be reduced when applying STM with the

deep CNN because it is still able to completely overfit on the frequency domain. One approach to test this hypothesis in a future work would be to compare and visualise the extracted features after applying the trained filters of a deep CNN with those of a shallow one [137;138]. Ultimately, a conclusion could be drawn if deeper CNN are more able to extract frequency domain specific features and consequently deeper models should be used when applying the Smooth Time Mask augmentation technique.

In comparison to the visualization approaches used by Rommel et al. [27], the box plots in combination with a point plot used to evaluate the effect of the applied augmentation techniques in this work do not depend on the assumption of Gaussian distributed results. Especially in an intra-subject setting, this method gave an intuition over the real distribution of the results and showed, how a certain augmentation technique affects the result distribution over the different subjects. Additionally, it has shown that the average is less expressive metric when assessing an augmentation technique in an intra-subject scenario with a small number of subjects and rather the median should be considered. This stems from the observation that qualitative subjects often profit more from an augmentation, while the data for difficult subjects gets even more challenging for the models to discriminate on. In a future work, an additional method to visualise the effects of a certain augmentation techniques, or to check its validity, is the use of topographical maps [113;28]. Especially for methods which effect the spatial domain like Channels Dropout. The maps could be used to investigate further on inter-subject variabilities or to find a reason why certain subjects are more difficult for the model. In the field of data augmentation, however, topographic maps are mainly used in the generation of own synthetic data with generative models such as Variational Autoencoders (VAE) or Generative Adversarial Networks (GAN) [28;139;140;141].

6.2 Future Work

This section will prioritise and summarise possible future work that has been taken up from the discussions in this work. As a first proposed work package, a second LOOCV loop should be added for the inter-subject experiments, so that each subject builds the test set once. To address the variability between subjects, then an Euclidean alignment in the preprocessing step is proposed. Furthermore, additional feature extraction methods need to be integrated during preprocessing for the SVM or traditional machine learning models in general. Namely, it is suggested to implement a feature extraction based on the wavelet transform, as it is able to retain information of the time as well as for the frequency domain and is also more suitable for non-stationary processes than methods based on the Fourier transform.

With second priority the experiments should be even more expanded. This encompasses to perform the experiments on a third data set to obtain a more confident statement of the trends observed. Because the deep CNN still has limited capacity by recent standards, an extension of the experiments with an additional model is advised, which further contrasts the results of the shallow CNN. For instance, promising results regarding inter-subject transferability could be achieved with inception based models like proposed in the work of Santamarìa et al. [91]. Also in the second work package, it is advised to generate additional visualisations to evaluate the assessed techniques even further. Firstly, topographic maps can be added to assess the validity of the augmentation techniques applied. Secondly, the EEG signal could be visualised after applying the convolutional filters to assess on which feature characteristics the model focuses on and to find an explanation for what distinguishes the subjects with good model performance from those with poor model performance.

Finally, in a last package, it is proposed to generate own synthetic data. As introduced in the discussion, there are existing papers which use VAE or GAN for synthetic data generation, however, often for emotion detection classification tasks. Nevertheless, the methods and models applied would need to be selected carefully to not overlap with existing works. Also, when considering the relative increase in accuracy in the referenced papers [28;139;140;141], often only comparable improvements of 1-6% could be achieved as with transformation-based methods. Although such a direct comparison cannot be made and should be considered with caution, it can still give a sense of what to expected. Finally, it could also mean that there is still room for improvement when it comes to synthetically generated data augmentation.

Chapter 7

Directories

Bibliography

[1] Yannick Roy, Hubert Banville, Isabela Albuquerque, Alexandre Gramfort, Tiago H. Falk, and Jocelyn Faubert. Deep learning-based electroencephalography analysis: A systematic review. *Journal of Neural Engineering*, 16, 8 2019. ISSN 17412552. doi: 10.1088/1741-2552/ab260c.

[2] U. Rajendra Acharya, Yuki Hagiwara, Sunny Nitin Deshpande, S. Suren, Joel En Wei Koh, Shu Lih Oh, N. Arunkumar, Edward J. Ciaccio, and Choo Min Lim. Characterization of focal eeg signals: A review. *Future Generation Computer Systems*, 91:290–299, 2 2019. ISSN 0167739X. doi: 10.1016/J.FUTURE.2018.08.044.

[3] Iñaki Iturrate, Ricardo Chavarriaga, and José del R. Millán. General principles of machine learning for brain-computer interfacing. *Handbook of Clinical Neurology*, 168:311–328, 1 2020. ISSN 0072-9752. doi: 10.1016/B978-0-444-63934-9.00023-8.

[4] Robert, Serafeim Perdikis, Luca Tonin, Andrea Biasiucci, Michele Tavella, Marco Creatura, Alberto Molina, Abdul Al-Khodairy, Tom Carlson, and José d.R. Millán. Transferring brain–computer interfaces beyond the laboratory: Successful application control for motor-disabled users. *Artificial Intelligence in Medicine*, 59:121–132, 10 2013. ISSN 0933-3657. doi: 10.1016/J.ARTMED.2013.08.004. URL `https://www.sciencedirect.com/science/article/pii/S0933365713001218?via%3Dihub`.

[5] Reinhold Scherer and Carmen Vidaurre. Motor imagery based brain-computer interfaces. *Smart Wheelchairs and Brain-computer Interfaces: Mobile Assistive Technologies*, pages 171–195, 1 2018. doi: 10.1016/B978-0-12-812892-3.00008-X.

[6] Martin Lotze and Ulrike Halsband. Motor imagery. *Journal of physiology, Paris*, 99:386–395, 6 2006. ISSN 0928-4257. doi: 10.1016/j.jphysparis.2006.03.012. Place: France.

[7] Hamdi Altaheri, Ghulam Muhammad, Mansour Alsulaiman, Syed Umar Amin, Ghadir Ali Altuwaijri, Wadood Abdul, Mohamed A Bencherif, and Mohammed Faisal. Deep learning techniques for classification of electroencephalogram (eeg) motor imagery (mi) signals: a review. *Neural Computing and Applications*, 8 2021. ISSN 1433-3058. doi: 10.1007/s00521-021-06352-5. URL `https://doi.org/10.1007/s00521-021-06352-5`.

[8] Alexander Craik, Yongtian He, and Jose L Contreras-Vidal. Deep learning for electroencephalogram (eeg) classification tasks: a review. *Journal of neural engineering*, 16:31001, 2019. Publisher: IOP Publishing.

[9] F Lotte, L Bougrain, A Cichocki, M Clerc, M Congedo, A Rakotomamonjy, and F Yger. A review of classification algorithms for eeg-based brain–computer interfaces: a 10 year update. *Journal of Neural Engineering*, 15:31005, 4 2018. doi: 10.1088/1741-2552/aab2f2. URL `https://dx.doi.org/10.1088/1741-2552/aab2f2`.

[10] Swati Aggarwal and Nupur Chugh. Review of machine learning techniques for eeg based brain computer interface. *Archives of Computational Methods in Engineering*, 29:3001–3020, 8 2022. ISSN 1886-1784. doi: 10.1007/s11831-021-09684-6. URL `https://doi.org/10.1007/s11831-021-09684-6`.

[11] Scott Cole and Bradley Voytek. Cycle-by-cycle analysis of neural oscillations. *bioRxiv*, 2018. doi: 10.1101/302000. URL https://www.biorxiv.org/content/early/2018/04/16/302000.

[12] A Gramfort, D Strohmeier, J Haueisen, M S Hämäläinen, and M Kowalski. Time-frequency mixed-norm estimates: sparse m/eeg imaging with non-stationary source activations. *NeuroImage*, 70: 410–422, 4 2013. ISSN 1095-9572 1053-8119. doi: 10.1016/j.neuroimage.2012.12.051. Place: United States.

[13] Mengxi Dai, Dezhi Zheng, Rui Na, Shuai Wang, and Shuailei Zhang. Eeg classification of motor imagery using a novel deep learning framework. *Sensors*, 19, 2019. ISSN 1424-8220. doi: 10.3390/s19030551. URL https://www.mdpi.com/1424-8220/19/3/551.

[14] Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, R Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. volume 2. Morgan-Kaufmann, 1989. URL https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf.

[15] Yang Li, Xian-Rui Zhang, Bin Zhang, Meng-Ying Lei, Wei-Gang Cui, and Yu-Zhu Guo. A channel-projection mixed-scale convolutional neural network for motor imagery eeg decoding. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 27:1170–1180, 2019. doi: 10.1109/TNSRE.2019.2915621.

[16] Nima Bigdely-Shamlo, Scott Makeig, and Kay A Robbins. Preparing laboratory and real-world eeg data for large-scale analysis: A containerized approach. *Frontiers in Neuroinformatics*, 10, 2016. ISSN 1662-5196. doi: 10.3389/fninf.2016.00007. URL https://www.frontiersin.org/articles/10.3389/fninf.2016.00007.

[17] David A van Dyk and Xiao-Li Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10:1–50, 2001. doi: 10.1198/10618600152418584. URL https://doi.org/10.1198/10618600152418584.

[18] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6:60, 7 2019. ISSN 2196-1115. doi: 10.1186/s40537-019-0197-0. URL https://doi.org/10.1186/s40537-019-0197-0.

[19] Justus T C Schwabedal, John C Snyder, Ayse Cakmak, Shamim Nemati, and Gari D Clifford. Addressing class imbalance in classification problems of noisy signals by using fourier transform surrogates, 2018. URL https://arxiv.org/abs/1806.08675.

[20] Aaqib Saeed, David Grangier, Olivier Pietquin, and Neil Zeghidour. Learning from heterogeneous eeg signals with differentiable channel reordering, 2020. URL https://arxiv.org/abs/2010.13694.

[21] Fang Wang, Sheng-Hua Zhong, Jianfeng Peng, Jianmin Jiang, and Yan Liu. Data augmentation for eeg-based emotion recognition with deep convolutional neural networks. 2018. doi: 10.1007/978-3-319-73600-6_8. URL https://doi.org/10.1007/978-3-319-73600-6_8.

[22] Olivier Deiss, Siddharth Biswal, Jing Jin, Haoqi Sun, M Brandon Westover, and Jimeng Sun. Hamlet: Interpretable human and machine co-learning technique. 2018.

[23] Mostafa Neo Mohsenvand, Mohammad Rasool Izadi, and Pattie Maes. Contrastive representation learning for electroencephalogram classification. pages 238–253. PMLR, 2020. URL http://proceedings.mlr.press/v136/mohsenvand20a.html.

[24] Joseph Y Cheng, Hanlin Goh, Kaan Dogrusoz, Oncel Tuzel, and Erdrin Azemi. Subject-aware contrastive learning for biosignals, 2020. URL https://arxiv.org/abs/2007.04871.

[25] Mario Michael Krell and Su-Kyoung Kim. Rotational data augmentation for electroencephalographic data. volume 2017, pages 471–474, 11 2017. doi: 10.1109/EMBC.2017.8036864.

[26] Hongyi Zhang, Moustapha Cissé, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017. URL http://arxiv.org/abs/1710.09412.

[27] Cédric Rommel, Joseph Paillard, Thomas Moreau, and Alexandre Gramfort. Data augmentation for learning predictive models on eeg: a systematic comparison, 2022. URL https://arxiv.org/abs/2206.14483.

[28] Olawunmi George, Roger Smith, Praveen Madiraju, Nasim Yahyasoltani, and Sheikh Iqbal Ahamed. Data augmentation strategies for eeg-based motor imagery decoding. *Heliyon*, 8, 8 2022. ISSN 24058440. doi: 10.1016/J.HELIYON.2022.E10240.

[29] Elnaz Lashgari, Dehua Liang, and Uri Maoz. Invited review data augmentation for deep-learning-based electroencephalography. 2020. doi: 10.1016/j.jneumeth.2020.108885. URL https://doi.org/10.1016/j.jneumeth.2020.108885.

[30] Paulo Henrique Gubert, Márcio Holsbach Costa, Cleison Daniel Silva, and Alexande Trofino-Neto. The performance impact of data augmentation in csp-based motor-imagery systems for bci applications. *Biomedical Signal Processing and Control*, 62:102152, 2020. ISSN 1746-8094. doi: https://doi.org/10.1016/j.bspc.2020.102152. URL https://www.sciencedirect.com/science/article/pii/S1746809420302974.

[31] Weijian Huang, Li Wang, Zhenxiong Yan, and Yanjun Liu. Classify motor imagery by a novel cnn with data augmentation. pages 192–195, 2020. doi: 10.1109/EMBC44109.2020.9176361.

[32] Zhiwen Zhang, Feng Duan, Jordi Solé-Casals, Josep Dinarès-Ferran, Andrzej Cichocki, Zhenglu Yang, and Zhe Sun. A novel deep learning approach with data augmentation to classify motor imagery signals. *IEEE Access*, PP:1, 11 2019. doi: 10.1109/ACCESS.2019.2895133.

[33] Josep Dinarès-Ferran, Rupert Ortner, Christoph Guger, and Jordi Solé-Casals. A new method to generate artificial frames using the empirical mode decomposition for an eeg-based motor imagery bci. *Frontiers in Neuroscience*, 12, 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00308. URL https://www.frontiersin.org/articles/10.3389/fnins.2018.00308.

[34] Haider Alwasiti and Mohd Zuki Yusoff. Motor imagery classification for brain computer interface using deep convolutional neural networks and mixup augmentation. *IEEE Open Journal of Engineering in Medicine and Biology*, 3:171–177, 2022. doi: 10.1109/OJEMB.2022.3220150.

[35] Daniel Freer and Guang-Zhong Yang. Data augmentation for self-paced motor imagery classification with c-lstm. *Journal of Neural Engineering*, 17:16041, 1 2020. doi: 10.1088/1741-2552/ab57c0. URL https://doi.org/10.1088/1741-2552/ab57c0.

[36] Robin Tibor Schirrmeister, Jost Tobias Springenberg, Lukas Dominique Josef Fiederer, Martin Glasstetter, Katharina Eggensperger, Michael Tangermann, Frank Hutter, Wolfram Burgard, and Tonio Ball. Deep learning with convolutional neural networks for eeg decoding and visualization. *Human Brain Mapping*, 38:5391–5420, 2017. doi: https://doi.org/10.1002/hbm.23730. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/hbm.23730.

[37] Robert Leeb, Felix Lee, Claudia Keinrath, Reinhold Scherer, Horst Bischof, and Gert Pfurtscheller. Brain–computer communication: Motivation, aim, and impact of exploring a virtual apartment. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 15:473–482, 2007. doi: 10.1109/TNSRE.2007.906956.

[38] Ary L Goldberger, Luis A N Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101:e215–e220, 2000. Publisher: Am Heart Assoc.

[39] Siuly Siuly, Yan Li, and Yanchun Zhang. Eeg signal analysis and classification. *IEEE Trans Neural Syst Rehabilit Eng*, 11:141–144, 2016. Publisher: Springer.

[40] Neep Hazarika, Jean Zhu Chen, Ah Chung Tsoi, and Alex Sergejew. Classification of eeg signals using the wavelet transform. *Signal Processing*, 59:61–72, 1997. ISSN 0165-1684. doi: https://doi.org/10.1016/S0165-1684(97)00038-8. URL https://www.sciencedirect.com/science/article/pii/S0165168497000388.

[41] Hojjat Adeli, Ziqin Zhou, and Nahid Dadmehr. Analysis of eeg records in an epileptic patient using wavelet transform. *Journal of neuroscience methods*, 123:69–87, 2 2003. ISSN 0165-0270. doi: 10.1016/s0165-0270(02)00340-0. Place: Netherlands.

[42] Michal Teplan. Fundamentals of eeg measurement. *Measurement science review*, 2:1–11, 2002.

[43] D Puthankattil Subha, Paul K Joseph, Rajendra Acharya U, and Choo Min Lim. Eeg signal analysis: A survey. *Journal of Medical Systems*, 34:195–212, 4 2010. ISSN 1573-689X. doi: 10.1007/s10916-008-9231-z. URL https://doi.org/10.1007/s10916-008-9231-z.

[44] Nima Bigdely-Shamlo, Tim Mullen, Christian Kothe, Kyung-Min Su, and Kay A Robbins. The prep pipeline: standardized preprocessing for large-scale eeg analysis. *Frontiers in Neuroinformatics*, 9, 2015. ISSN 1662-5196. doi: 10.3389/fninf.2015.00016. URL https://www.frontiersin.org/articles/10.3389/fninf.2015.00016.

[45] Mainak Jas, Denis A Engemann, Yousra Bekhti, Federico Raimondo, and Alexandre Gramfort. Autoreject: Automated artifact rejection for meg and eeg data. *NeuroImage*, 159:417–429, 10 2017. ISSN 1095-9572 1053-8119. doi: 10.1016/j.neuroimage.2017.06.030. Place: United States.

[46] Guy P Nason. Stationary and non-stationary time series. *Statistics in volcanology*, 60, 2006. Publisher: Geological Society of London, London.

[47] Maureen Clerc, Laurent Bougrain, and Fabien Lotte. *Brain-computer interfaces 1: Methods and perspectives*. John Wiley & Sons, 2016.

[48] Jun Ma, Banghua Yang, Wenzheng Qiu, Yunzhe Li, Shouwei Gao, and Xinxing Xia. A large EEG dataset for studying cross-session variability in motor imagery brain-computer interface. *Scientific Data*, 9(1):531, September 2022. ISSN 2052-4463. doi: 10.1038/s41597-022-01647-1. URL https://doi.org/10.1038/s41597-022-01647-1.

[49] Hohyun Cho, Minkyu Ahn, Sangtae Ahn, Moonyoung Kwon, and Sung Chan Jun. EEG datasets for motor imagery brain–computer interface. *GigaScience*, 6(7), May 2017. ISSN 2047-217X. doi: 10.1093/gigascience/gix034. URL https://doi.org/10.1093/gigascience/gix034. _eprint: https://academic.oup.com/gigascience/article-pdf/6/7/gix034/25515099/gix034.pdf.

[50] Michael Tangermann, Klaus-Robert Müller, Ad Aertsen, Niels Birbaumer, Christoph Braun, Clemens Brunner, Robert Leeb, Carsten Mehring, Kai J Miller, Gernot R Müller-Putz, Guido Nolte, Gert Pfurtscheller, Hubert Preissl, Gerwin Schalk, Alois Schlögl, Carmen Vidaurre, Stephan Waldert, and Benjamin Blankertz. Review of the bci competition iv. *Frontiers in neuroscience*, 6:55, 2012. ISSN 1662-453X 1662-4548. doi: 10.3389/fnins.2012.00055.

[51] Josef Faller, Carmen Vidaurre, Teodoro Solis-Escalante, Christa Neuper, and Reinhold Scherer. Autocalibration and recurrent adaptation: towards a plug and play online erd-bci. *IEEE transactions on neural systems and rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society*, 20:313–319, 5 2012. ISSN 1558-0210 1534-4320. doi: 10.1109/TNSRE.2012.2189584. Place: United States.

[52] Riitta Hari and Aina Puce. *MEG-EEG Primer*. Oxford University Press, 10 2017. ISBN 9780190497774. doi: 10.1093/med/9780190497774.001.0001. URL https://doi.org/10.1093/med/9780190497774.001.0001.

[53] Shen Luo and Paul Johnston. A review of electrocardiogram filtering. *Journal of Electrocardiology*, 43:486–496, 2010. ISSN 0022-0736. doi: https://doi.org/10.1016/j.jelectrocard.2010.07.007. URL https://www.sciencedirect.com/science/article/pii/S0022073610002852.

[54] Emanuele Maiorana, Jordi Solé-Casals, and Patrizio Campisi. Eeg signal preprocessing for biometric recognition. *Machine Vision and Applications*, 27:1351–1360, 11 2016. ISSN 1432-1769. doi: 10.1007/s00138-016-0804-4. URL https://doi.org/10.1007/s00138-016-0804-4.

[55] Nelly Elsayed, Zaghloul Saad Zaghloul, and Magdy Bayoumi. Brain computer interface: Eeg signal preprocessing issues and solutions. *Int. J. Comput. Appl*, 169:975–8887, 2017.

[56] M Rajya Lakshmi, T V Prasad, and Dr V Chandra Prakash. Survey on eeg signal processing methods. *International journal of advanced research in computer science and software engineering*, 4, 2014.

[57] Nik Khadijah Nik Aznan and Yeon-Mo Yang. Applying kalman filter in eeg-based brain computer interface for motor imagery classification. pages 688–690, 2013. doi: 10.1109/ICTC.2013.6675451.

[58] Alain de Cheveigné and Israel Nelken. Filters: When, why, and how (not) to use them. *Neuron*, 102:280–293, 2019. ISSN 0896-6273. doi: https://doi.org/10.1016/j.neuron.2019.02.039. URL https://www.sciencedirect.com/science/article/pii/S0896627319301746.

[59] E Huigen, A Peper, and C A Grimbergen. Investigation into the origin of the noise of surface electrodes. *Medical and Biological Engineering and Computing*, 40:332–338, 5 2002. ISSN 1741-0444. doi: 10.1007/BF02344216. URL https://doi.org/10.1007/BF02344216.

[60] Emily S Kappenman and Steven J Luck. The effects of electrode impedance on data quality and statistical significance in erp recordings. *Psychophysiology*, 47:888–904, 2010. doi: https://doi.org/10.1111/j.1469-8986.2010.01009.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-8986.2010.01009.x.

[61] Xiaoxi Kang, Dini Oktarina Dwi Handayani, and Hamwira Yaacob. Comparison between butterworth bandpass and stationary wavelet transform filter for electroencephalography signal. *IOP Conference Series: Materials Science and Engineering*, 1077:12024, 2 2021. doi: 10.1088/1757-899X/1077/1/012024. URL https://dx.doi.org/10.1088/1757-899X/1077/1/012024.

[62] S S Daud and R Sudirman. Butterworth bandpass and stationary wavelet transform filter comparison for electroencephalography signal. pages 123–126, 2015. doi: 10.1109/ISMS.2015.29.

[63] Gabriele Gratton. Dealing with artifacts: The eog contamination of the event-related brain potential. *Behavior Research Methods, Instruments and Computers*, 30:44–53, 1998. Publisher: Springer.

[64] Rowan McAllister. Data-efficient policy search using pilco and directed-exploration. 2016.

[65] Philip S Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. *CoRR*, abs/1604.00923, 2016. URL http://arxiv.org/abs/1604.00923. arXiv: 1604.00923.

[66] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learningfor image recognition. *ComputerScience*, 2015.

[67] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. pages 2818–2826, 2016.

[68] Andrew Ng. *Machine Learning Yearning*. Online Draft, 2017. URL http://www.mlyearning.org/,/bib/ng/ng2017mlyearning/Ng_MLY01_13.pdf.

[69] Simanto Saha and Mathias Baumert. Intra- and inter-subject variability in eeg-based sensorimotor brain computer interface: A review. *Frontiers in Computational Neuroscience*, 13, 2020. ISSN 1662-5188. doi: 10.3389/fncom.2019.00087. URL https://www.frontiersin.org/articles/10.3389/fncom.2019.00087.

[70] Chao Tang, Yunhuan Li, and Badong Chen. Comparison of cross-subject eeg emotion recognition algorithms in the bci controlled robot contest in world robot contest 2021. *Brain Science Advances*, 8:142–152, 2022. doi: 10.26599/BSA.2022.9050013. URL https://doi.org/10.26599/BSA.2022.9050013.

[71] Javier Fdez, Nicholas Guttenberg, Olaf Witkowski, and Antoine Pasquali. Cross-subject eeg-based emotion recognition through neural networks with stratified normalization. *Frontiers in Neuroscience*, 15, 2021. ISSN 1662-453X. doi: 10.3389/fnins.2021.626277. URL https://www.frontiersin.org/articles/10.3389/fnins.2021.626277.

[72] Tali M Ball, Lindsay M Squeglia, Susan F Tapert, and Martin P Paulus. Double dipping in machine learning: Problems and solutions. *Biological psychiatry. Cognitive neuroscience and neuroimaging*, 5:261–263, 3 2020. ISSN 2451-9030 2451-9022. doi: 10.1016/j.bpsc.2019.09.003. Place: United States.

[73] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8, 2019. ISSN 2079-9292. doi: 10.3390/electronics8080832. URL https://www.mdpi.com/2079-9292/8/8/832.

[74] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning, 2017. URL https://arxiv.org/abs/1702.08608.

[75] Brendan Juba and Hai S Le. Precision-recall versus accuracy and the role of large data sets. volume 33, pages 4039–4048, 2019. Issue: 01.

[76] I Bratko. Machine learning: Between accuracy and interpretability. pages 163–177. Springer Vienna, 1997. ISBN 978-3-7091-2668-4.

[77] Charles X Ling, Jin Huang, and Harry Zhang. Auc: A better measure than accuracy in comparing learning algorithms. pages 329–341. Springer Berlin Heidelberg, 2003. ISBN 978-3-540-44886-0.

[78] Jason Brownlee. Failure of classification accuracy for imbalanced class distributions, 1 2021. URL https://machinelearningmastery.com/failure-of-accuracy-for-imbalanced-class-distributions/.

[79] Classification: Accuracy, 7 2022. URL https://developers.google.com/machine-learning/crash-course/classification/accuracy.

[80] F Pedregosa, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot, and E Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[81] Classification: Precision and recall, 7 2022. URL https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall.

[82] Olivier Caelen. A bayesian interpretation of the confusion matrix. *Annals of Mathematics and Artificial Intelligence*, 81:429–450, 12 2017. ISSN 1573-7470. doi: 10.1007/s10472-017-9564-8. URL https://doi.org/10.1007/s10472-017-9564-8.

[83] David M W Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2020. doi: 10.48550/ARXIV.2010.16061. URL https://arxiv.org/abs/2010.16061. Publisher: arXiv.

[84] Zhe Hui Hoo, Jane Candlish, and Dawn Teare. What is an roc curve?, 2017. Issue: 6 Pages: 357–359 Publication Title: Emergency Medicine Journal Volume: 34.

[85] Luzia Gonçalves, Ana Subtil, M Rosário Oliveira, and Patricia de Zea Bermudez. Roc curve estimation: An overview. *REVSTAT-Statistical journal*, 12:1–20, 2014.

[86] Arie Ben-David. About the relationship between roc curves and cohen's kappa. *Engineering Applications of Artificial Intelligence*, 21:874–882, 9 2008. ISSN 09521976. doi: 10.1016/J.ENGAPPAI.2007.09.009.

[87] Ahnaf Rashik Hassan and Abdulhamit Subasi. Automatic identification of epileptic seizures from eeg signals using linear programming boosting. *Computer Methods and Programs in Biomedicine*, 136:65–77, 11 2016. ISSN 18727565. doi: 10.1016/J.CMPB.2016.08.013.

[88] Arnaud Sors, Stéphane Bonnet, Sébastien Mirek, Laurent Vercueil, and Jean François Payen. A convolutional neural network for sleep stage scoring from raw single-channel eeg. *Biomedical Signal Processing and Control*, 42:107–114, 4 2018. ISSN 17468108. doi: 10.1016/J.BSPC.2017.12.001.

[89] M Maclure and W C Willett. Misinterpretation and misuse of the kappa statistic. *American journal of epidemiology*, 126:161–169, 8 1987. ISSN 0002-9262. doi: 10.1093/aje/126.2.161. Place: United States.

[90] Zijie J Wang, Robert Turko, Omar Shaikh, Haekyu Park, Nilaksh Das, Fred Hohman, Minsuk Kahng, and Duen Horng Polo Chau. Cnn explainer: Learning convolutional neural networks with interactive visualization. *IEEE Transactions on Visualization and Computer Graphics*, 27: 1396–1406, 2021. doi: 10.1109/TVCG.2020.3030418.

[91] Eduardo Santamaría-Vázquez, Víctor Martínez-Cagigal, Fernando Vaquerizo-Villar, and Roberto Hornero. Eeg-inception: A novel deep convolutional neural network for assistive erp-based brain-computer interfaces. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 28: 2773–2782, 2020. doi: 10.1109/TNSRE.2020.3048106.

[92] J Karthika, M Ganesan, and R Lavanya. Combining cnn with autoencoder for motor-imagery classification. pages 1295–1300, 2022. doi: 10.1109/ICCES54183.2022.9835739.

[93] Kai Keng Ang, Zheng Yang Chin, Haihong Zhang, and Cuntai Guan. Filter bank common spatial pattern (fbcsp) in brain-computer interface. pages 2390–2397, 2008. doi: 10.1109/IJCNN.2008.4634130.

[94] Yijun Wang, Shangkai Gao, and Xiaorong Gao. Common spatial pattern method for channel selelction in motor imagery based brain-computer interface. pages 5392–5395. IEEE, 2006.

[95] Y-Lan Boureau, Nicolas Le Roux, Francis Bach, Jean Ponce, and Yann LeCun. Ask the locals: Multi-way local pooling for image recognition. pages 2651–2658, 2011. doi: 10.1109/ICCV.2011.6126555.

[96] Rahul Sharma, Minju Kim, and Akansha Gupta. Motor imagery classification in brain-machine interface with machine learning algorithms: Classical approach to multi-layer perceptron model. *Biomedical Signal Processing and Control*, 71, 1 2022. ISSN 17468108. doi: 10.1016/J.BSPC.2021.103101.

[97] Minmin Zheng, Banghua Yang, and Yunlong Xie. Eeg classification across sessions and across subjects through transfer learning in motor imagery-based brain-machine interface system. *Medical and Biological Engineering and Computing*, 58:1515–1528, 7 2020. ISSN 1741-0444. doi: 10.1007/s11517-020-02176-y. URL https://doi.org/10.1007/s11517-020-02176-y.

[98] Xuyang Zhu, Peiyang Li, Cunbo Li, Dezhong Yao, Rui Zhang, and Peng Xu. Separated channel convolutional neural network to realize the training free motor imagery bci systems. *Biomedical Signal Processing and Control*, 49:396–403, 2019. doi: 10.1016/j.bspc.2018.12.027. URL https://doi.org/10.1016/j.bspc.2018.12.027.

[99] Yuliang Ma, Xiaohui Ding, Qingshan She, Zhizeng Luo, Thomas Potter, and Yingchun Zhang. Classification of motor imagery eeg signals with support vector machines and particle swarm optimization. 2016. doi: 10.1155/2016/4941235. URL http://dx.doi.org/10.1155/2016/4941235.

[100] Alois Schlögl, Felix Lee, Horst Bischof, and Gert Pfurtscheller. Characterization of four-class motor imagery eeg data for the bci-competition 2005. *Journal of Neural Engineering*, 2:L14, 8 2005. doi: 10.1088/1741-2560/2/4/L02. URL https://dx.doi.org/10.1088/1741-2560/2/4/L02.

[101] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 9 1995. ISSN 1573-0565. doi: 10.1023/A:1022627411411. URL https://doi.org/10.1023/A:1022627411411.

[102] S Salcedo-Sanz, J L Rojo-Álvarez, M Martínez-Ramón, and G Camps-Valls. Support vector machines in engineering: an overview. *WIREs Data Mining and Knowledge Discovery*, 4:234–267, 2014. doi: https://doi.org/10.1002/widm.1125. URL https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1125.

[103] Richard G Brereton and Gavin R Lloyd. Support vector machines for classification and regression. *Analyst*, 135:230–267, 2010. doi: 10.1039/B918972F. URL http://dx.doi.org/10.1039/B918972F.

[104] Enzeng Dong, Changhai Li, Liting Li, Shengzhi Du, Abdelkader Nasreddine Belkacem, and Chao Chen. Classification of multi-class motor imagery with a novel hierarchical svm algorithm for brain–computer interfaces. *Medical and Biological Engineering and Computing*, 55:1809–1818, 10 2017. ISSN 1741-0444. doi: 10.1007/s11517-017-1611-4. URL https://doi.org/10.1007/s11517-017-1611-4.

[105] Sahar Selim, Manal Mohsen Tantawi, Howida A Shedeed, and Amr Badr. A csp-ba-svm approach for motor imagery bci system. *IEEE Access*, 6:49192–49208, 2018. doi: 10.1109/ACCESS.2018.2868178.

[106] Luke Taylor and Geoff Nitschke. Improving deep learning with generic data augmentation. pages 1542–1547, 2018. doi: 10.1109/SSCI.2018.8628742.

[107] Richard S Rosenberg and Steven Van Hout. The american academy of sleep medicine inter-scorer reliability program: sleep stage scoring. *Journal of clinical sleep medicine*, 9:81–87, 2013. Publisher: American Academy of Sleep Medicine.

[108] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15:1929–1958, 2014. Publisher: JMLR. org.

[109] Hubert Banville, Sean U.N. Wood, Chris Aimone, Denis Alexander Engemann, and Alexandre Gramfort. Robust learning from corrupted eeg with dynamic spatial filtering. *NeuroImage*, 251, 5 2022. ISSN 10959572. doi: 10.1016/J.NEUROIMAGE.2022.118994.

[110] Fernando da Silva, Jan Pieter Pijn, and Peter Boeijinga. Interdependence of eeg signals: Linear vs. nonlinear associations and the significance of time delays and phase shifts. *Brain Topography*, 2:9–18, 9 1989. ISSN 1573-6792. doi: 10.1007/BF01128839. URL https://doi.org/10.1007/BF01128839.

[111] Cédric Rommel, Thomas Moreau, Joseph Paillard, and Alexandre Gramfort. Cadda: Class-wise automatic differentiable data augmentation for eeg signals. *arXiv preprint arXiv:2106.13695*, 2021.

[112] Simon P. Kelly, Charles E. Schroeder, and Edmund C. Lalor. What does polarity inversion of extrastriate activity tell us about striate contributions to the early vep? a comment on ales et al. (2010). *NeuroImage*, 76:442–445, 8 2013. ISSN 1053-8119. doi: 10.1016/J.NEUROIMAGE.2012.03.081.

[113] R. C. Burgess and T. F. Collura. Polarity, localization, and field determination in electroencephalography. *The treatment of epilepsy: principles and practice. Philadelphia: Lea and Febiger*, pages 211–233, 1993.

[114] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. 10 2019.

[115] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout, 2017. URL https://arxiv.org/abs/1708.04552.

[116] Jayaram Vinary and Barachant Alexandre. Moabb: trustworthy algorithm benchmarking for bcis. *Journal of Neural Engineering*, 15, 9 2018. doi: 10.1088/1741-2552/aadea0. URL https://doi.org/10.1088/1741-2552/aadea0.

[117] Clemens Brunner, Robert Leeb, Gernot Müller-Putz, Alois Schlögl, and Gert Pfurtscheller. Bci competition 2008–graz data set a. *Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology*, 2008.

[118] Omry Yadan. Hydra - a framework for elegantly configuring complex applications, 2019. URL https://github.com/facebookresearch/hydra.

[119] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework, 2019. URL https://arxiv.org/abs/1907.10902.

[120] Lukas Biewald. Experiment tracking with weights and biases, 2020. URL https://www.wandb.com/. Software available from wandb.com.

[121] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[122] Marian Tietz, Thomas J Fan, Daniel Nouri, and Benjamin Bossan. Skorch: A scikit-learn compatible neural network library that wraps pytorch, 7 2017.

[123] Nicki Skafte Detlefsen, Jiri Borovec, Justus Schock, Ananya Harsh Jha, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin, and William Falcon. Torchmetrics - measuring reproducibility in pytorch. *Journal of Open Source Software*, 7:4101, 2022. doi: 10.21105/joss.04101. URL https://doi.org/10.21105/joss.04101. Publisher: The Open Journal.

[124] Zhanyuan Chang, Congcong Zhang, and Chuanjiang Li. Motor imagery eeg classification based on transfer learning and multi-scale convolution network. *Micromachines*, 13, 6 2022. ISSN 2072-666X. doi: 10.3390/mi13060927. Place: Switzerland.

[125] Lichao Xu, Minpeng Xu, Tzyy-Ping Jung, and Dong Ming. Review of brain encoding and decoding mechanisms for eeg-based brain-computer interface. *Cognitive neurodynamics*, 15:569–584, 8 2021. ISSN 1871-4080 1871-4099. doi: 10.1007/s11571-021-09676-z. Place: Netherlands.

[126] Vinay Jayaram, Morteza Alamgir, Yasemin Altun, Bernhard Scholkopf, and Moritz Grosse-Wentrup. Transfer learning in brain-computer interfaces. *IEEE Computational Intelligence Magazine*, 11:20–31, 2016. doi: 10.1109/MCI.2015.2501545.

[127] Paolo Zanini, Marco Congedo, Christian Jutten, Salem Said, and Yannick Berthoumieu. Transfer learning: A riemannian geometry framework with applications to brain–computer interfaces. *IEEE Transactions on Biomedical Engineering*, 65:1107–1116, 2018. doi: 10.1109/TBME.2017.2742541.

[128] He He and Dongrui Wu. Transfer learning for brain–computer interfaces: A euclidean space data alignment approach. *IEEE Transactions on Biomedical Engineering*, 67:399–410, 2020. doi: 10.1109/TBME.2019.2913914.

[129] Mary Judith Antony, Baghavathi Priya Sankaralingam, Rakesh Kumar Mahendran, Akber Abid Gardezi, Muhammad Shafiq, Jin-Ghoo Choi, and Habib Hamam. Classification of eeg using adaptive svm classifier with csp and online recursive independent component analysis. *Sensors (Basel, Switzerland)*, 22, 10 2022. ISSN 1424-8220. doi: 10.3390/s22197596. Place: Switzerland.

[130] C Vidaurre, T Jorajuría, A Ramos-Murguialday, K-R Müller, M Gómez, and V V Nikulin. Improving motor imagery classification during induced motor perturbations. *Journal of Neural Engineering*, 18:0460b1, 7 2021. doi: 10.1088/1741-2552/ac123f. URL https://dx.doi.org/10.1088/1741-2552/ac123f.

[131] Hardik Meisheri, Nagraj Ramrao, and Suman Mitra. Multiclass common spatial pattern for eeg based brain computer interface with adaptive learning classifier, 2018. URL https://arxiv.org/abs/1802.09046.

[132] Nitendra Kumar, Khursheed Alam, and Abul Hasan Siddiqi. Wavelet transform for classification of eeg signal using svm and ann. *Biomedical and Pharmacology Journal*, 10:2061–2069, 2017.

[133] Rajdeep Chatterjee, Tathagata Bandyopadhyay, and Debarshi Kumar Sanyal. Effects of wavelets on quality of features in motor-imagery eeg signal classification. pages 1346–1350, 2016. doi: 10.1109/WiSPNET.2016.7566356.

[134] Souvik Phadikar, Nidul Sinha, and Rajdeep Ghosh. Unsupervised feature extraction with autoencoders for eeg based multiclass motor imagery bci. *Expert Systems with Applications*, 213:118901, 2023. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2022.118901. URL https://www.sciencedirect.com/science/article/pii/S0957417422019194.

[135] Rupesh Mahamune and Shahedul H Laskar. Classification of the four-class motor imagery signals using continuous wavelet transform filter bank-based two-dimensional images. *International Journal of Imaging Systems and Technology*, 31:2237–2248, 2021. Publisher: Wiley Online Library.

[136] Yaojie, Xu Rong, Liu Guangming, Wang Xiaofan, Gong Yijing Luo Jing, and Wang. Channel drop out: A simple way to prevent cnn from overfitting in motor imagery based bci. pages 443–452. Springer Nature Singapore, 2021. ISBN 978-981-16-5940-9.

[137] Ali Nouri and Kyanoosh Azizi. Introducing a convolutional neural network and visualization of its filters for classification of eeg signal for ssvep task. *Frontiers in Biomedical Technologies*, 7: 151–159, 2020.

[138] Orestis Tsinalis, Paul M Matthews, Yike Guo, and Stefanos Zafeiriou. Automatic sleep stage scoring with single-channel eeg using convolutional neural networks, 2016. URL https://arxiv.org/abs/1610.01683.

[139] Guangcheng Bao, Bin Yan, Li Tong, Jun Shu, Linyuan Wang, Kai Yang, and Ying Zeng. Data augmentation for eeg-based emotion recognition using generative adversarial networks. *Frontiers in Computational Neuroscience*, 15, 2021. ISSN 1662-5188. doi: 10.3389/fncom.2021.723843. URL https://www.frontiersin.org/articles/10.3389/fncom.2021.723843.

[140] Yun Luo, Li-Zhen Zhu, Zi-Yu Wan, and Bao-Liang Lu. Data augmentation for enhancing eeg-based emotion recognition with deep generative models, 2020. URL https://arxiv.org/abs/2006.05331.

[141] Fatemeh Fahimi, Strahinja Dosen, Kai Keng Ang, Natalie Mrachacz-Kersting, and Cuntai Guan. Generative adversarial networks-based data augmentation for brain–computer interface. *IEEE Transactions on Neural Networks and Learning Systems*, 32:4039–4051, 2021. doi: 10.1109/TNNLS.2020.3016666.

List of Figures

List of Tables

List of Abbreviations

Abbr	Abbreviation
AI	Artificial Intelligence
API	Application Programming Interface
AUC	Area Under The Curve
BA	Bat Algorithm
BCI	Brain Computer Interface
CD	Channels Dropout
CNN / ConvNet	Convolutional Neural Network
CSP	Common Spatial Patterns
CV	Cross Validation
DL	Deep Learning
EA	Euclidean Alignment
EEG	Electroencephalography / Electroencephalogram
FB	Filter Bank
FBCSP	Filter Bank Common Spatial Patterns
FFT	Fast Fourier Transform
FPR	False Positive Rate
FT	Fourier Transform
FTS / FT Surrogate	Fourier Transform Surrogate
GAN	Generative Adversarial Network
HSVM	Hierarchical Support Vector Machine
KNN	K-Nearest Neighbors
LDA	Linear Discriminant Analysis
LOOCV	Leave One Out Cross Validation
MEG	Magnetoencephalography
MI	Motor Imagery
ML	Machine Learning
MLP	Multilayer Perceptron
PSD	Power Spectral Density
PSO	Particle Swarm Optimization
RA	Riemannian Alignment
RF	Random Forest
ROC	Receiver Operating Characteristic
STFT	Short Time Fourier Transform
STIM	Stimulus
STM	Smooth Time Mask
SVM	Support Vector Machine
TPR	True Positive Rate
VAE	Variational Autoencoder
WandB	Weights & Biases
WT	Wavelet Transform

Appendix A

Appendix

A.1 Data Set Comparison

Figure A.1: Data Set Comparison

A.2 Experiments

A.2.1 Preprocessing Hyperparameter Configuration

In table A.1 the hyperparameters used in the preprocessing step are visualised in favour of reproducibility. The names of the parameters are related to the implementation in Braindecode [36].

Preprocessing Step	Function Name	Hyperparameter Configuration
Channel Type Selection	pick_types	eeg: *True* meg: *False* stim: *False*
Scaling	scale	scale_factor: *1e6* apply_on_array: *True*
Bandpass Filter	filter	l_freq: *4 Hz* h_freq: *38 Hz*
Standardisation	exponential_moving_standardize	factor_new: *1e-3* init_block_size: *Data set window size*
Create Windows	create_windows_from_events	trial_start_offset_samples: *-0.5s* trial_stop_offset_samples: *0* window_size_samples: *Data set window size* window_stride_samples: *calculated* *n predictions per input depending* *on model output shape* drop_last_window: *False*

Table A.1: Preprocessing Hyperparameter Configuration

A.2.2 Experiments Plan

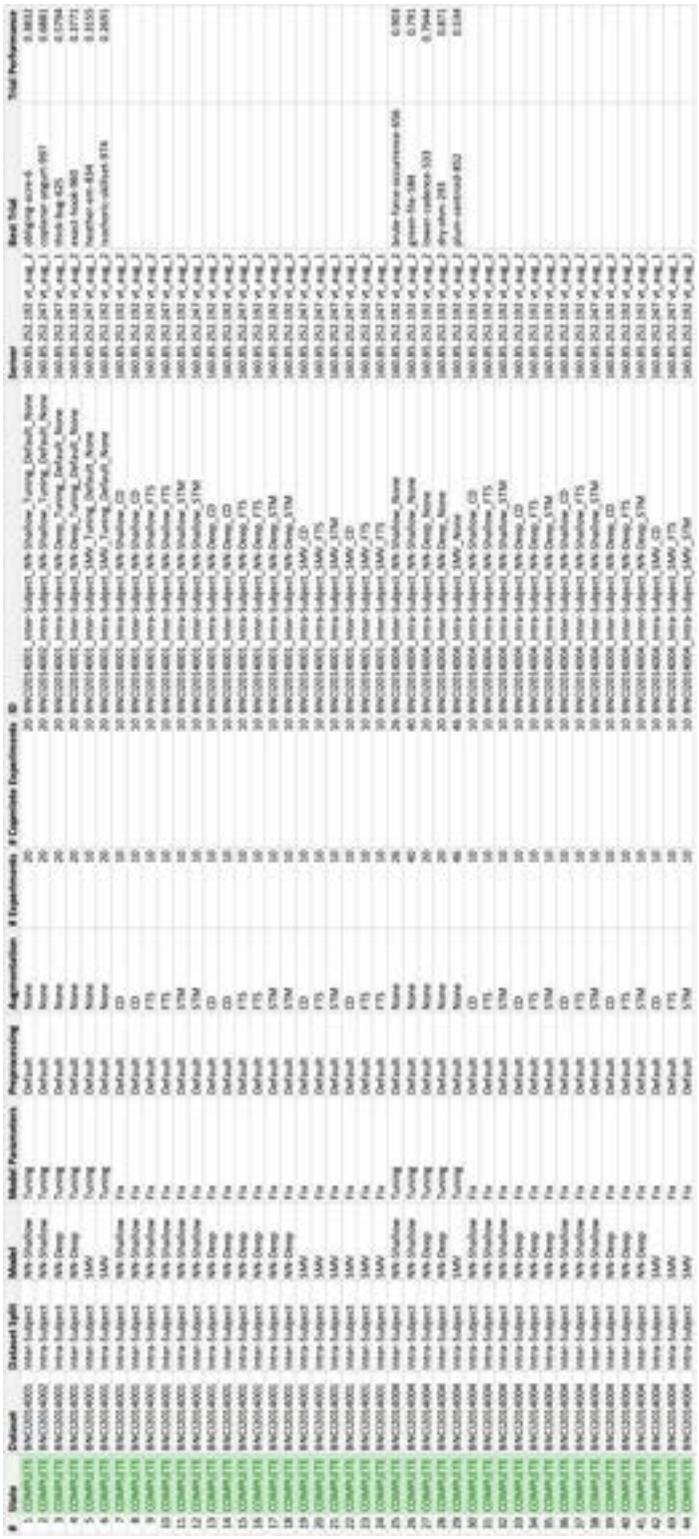


Figure A.2: Experiment Plan

A.2.3 Baseline Experiment Evaluation

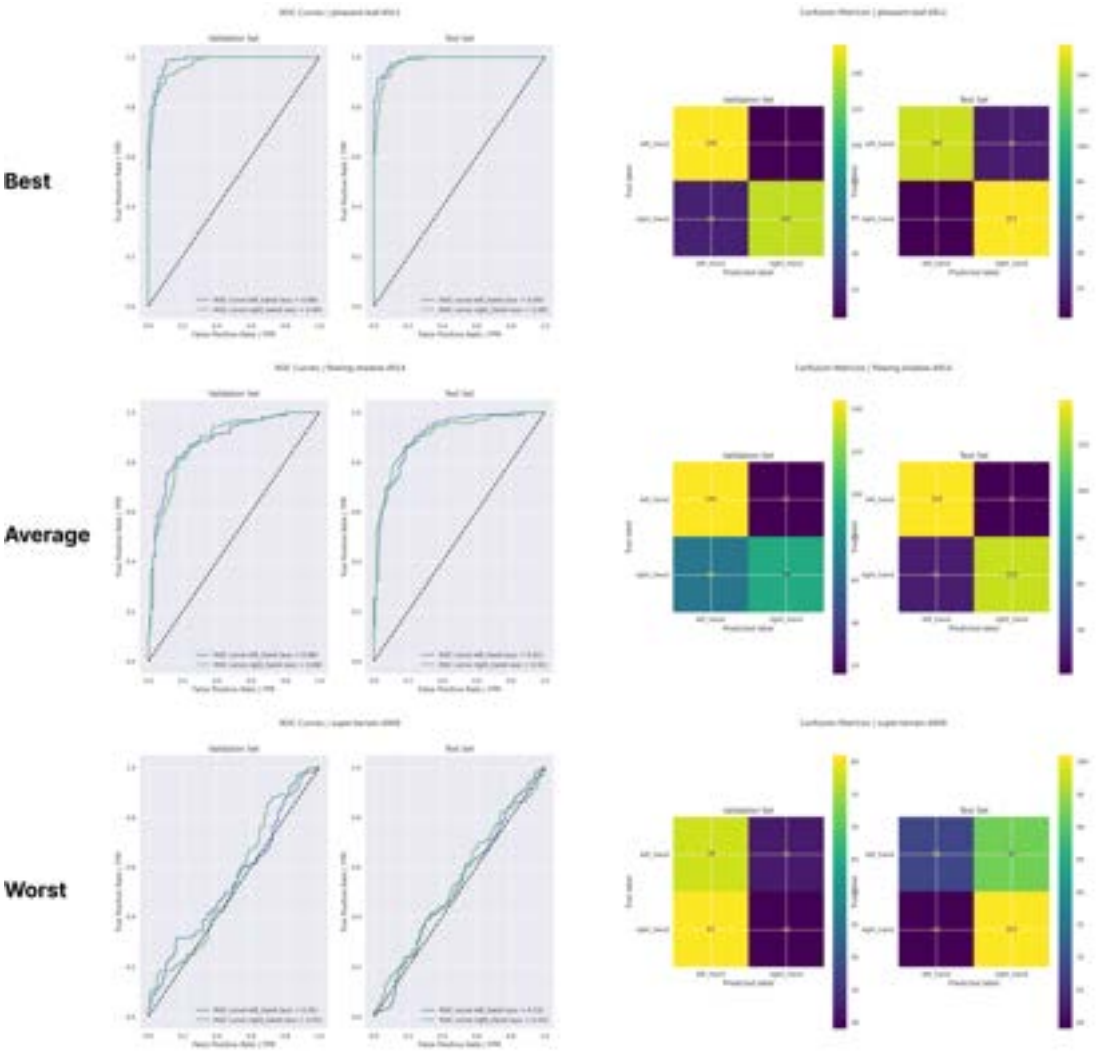


Figure A.3: Intra-Subject Baseline Experiment Performance Evaluation Shallow CNN on BNCI2014004

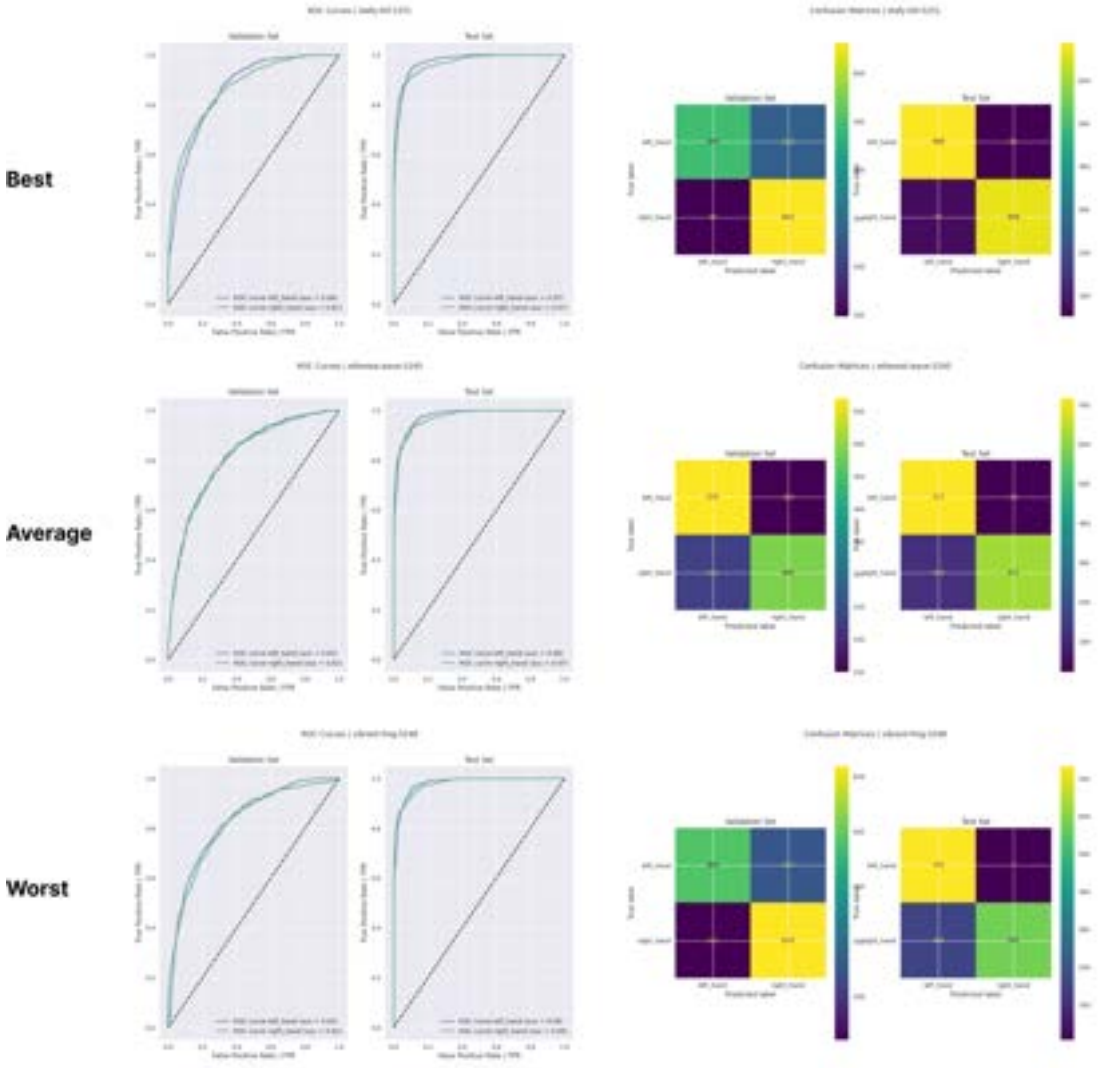


Figure A.4: Inter-Subject Baseline Experiment Performance Evaluation Shallow CNN on BNCI2014004

A.2.4 Learning Curves

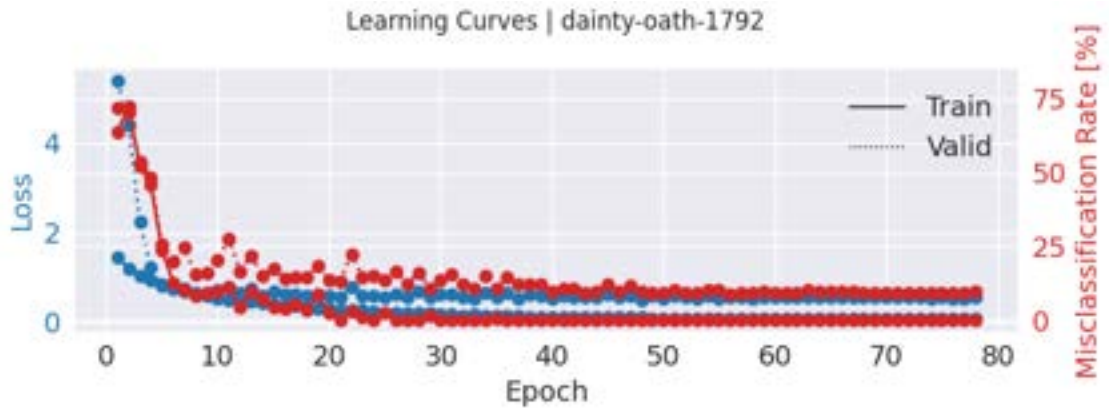


Figure A.5: Learning Curves | Shallow CNN | BNCI2014001 | Intra-Subject | Baseline

A.2.5 Variance Analysis

An example of a conducted variance analysis for Channels Dropout applied on the BNCI2014004 data set in an intra-subject training scenario using the shallow CNN.

Variance plot

Figure A.6 shows a scatter plot with the variance calculated by $train\ accuracy - test\ accuracy$ on the Y-axis and the hyperparameter p_{drop} on the X-axis. The colour is set dependent on the distinct subject.

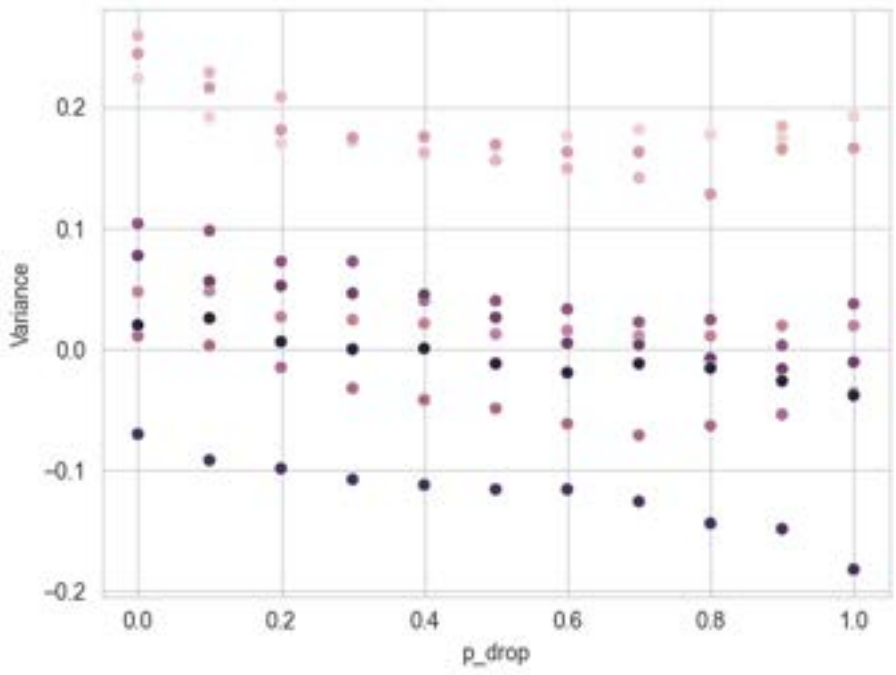


Figure A.6: Variance Plot | Channels Dropout | Shallow CNN | BNCI2014004 | Intra-Subject

Box plot

Figure A.7 shows subject-specific box plots to examine the distribution of variance reduction between subjects when increasing p_{drop}.

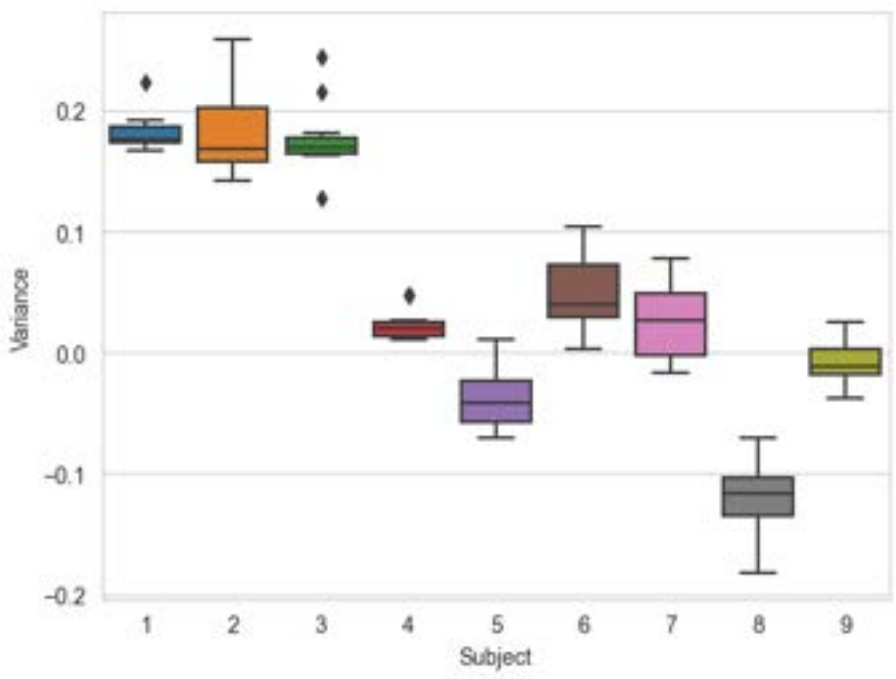


Figure A.7: Combined Scatter Plot | Channels Dropout | Shallow CNN | BNCI2014004 | Intra-Subject

Subject Specific Variance Plot

Figure A.8 shows the impact on variance when increasing the hyperparameter p_{drop} for each subject in the BNCI2014004 data set.

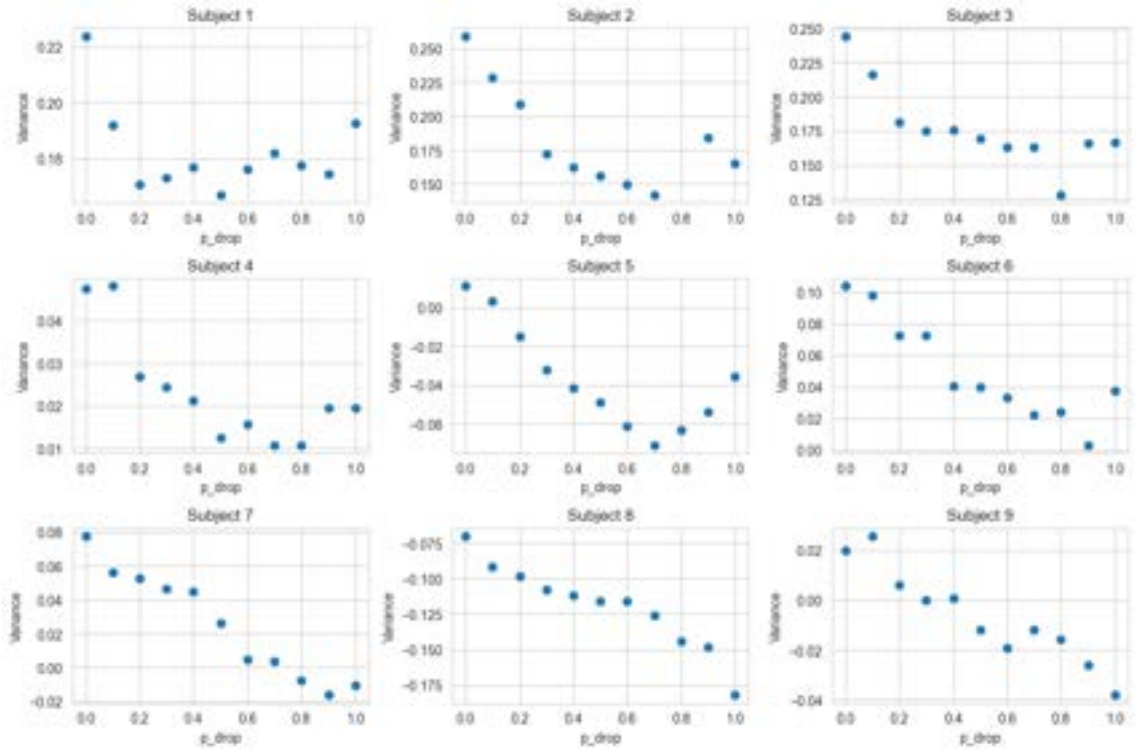


Figure A.8: Subject Specific Variance Plots | Channels Dropout | Shallow CNN | BNCI2014004 | Intra-Subject

Correlation

Figure A.9 shows the correlation of the variance to the hyperparameter p_{drop}. In addition, a box plot is generated to see the distribution of the correlations between the subjects.

Subject	Pearson Correlation	Spearman Correlation
1	-0.358	-0.009
2	-0.702	-0.648
3	-0.785	-0.836
4	-0.767	-0.758
5	-0.785	-0.745
6	-0.898	-0.897
7	-0.977	-0.991
8	-0.954	-0.998
9	-0.954	-0.936

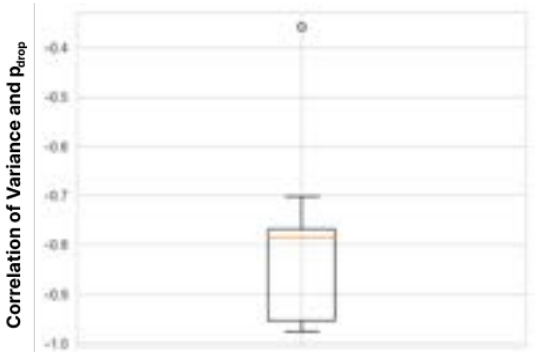


Figure A.9: Correlation | Channels Dropout | Shallow CNN | BNCI2014004 | Intra-Subject

A.3 Project Management

Figure A.10 shows the used project plan of this work.

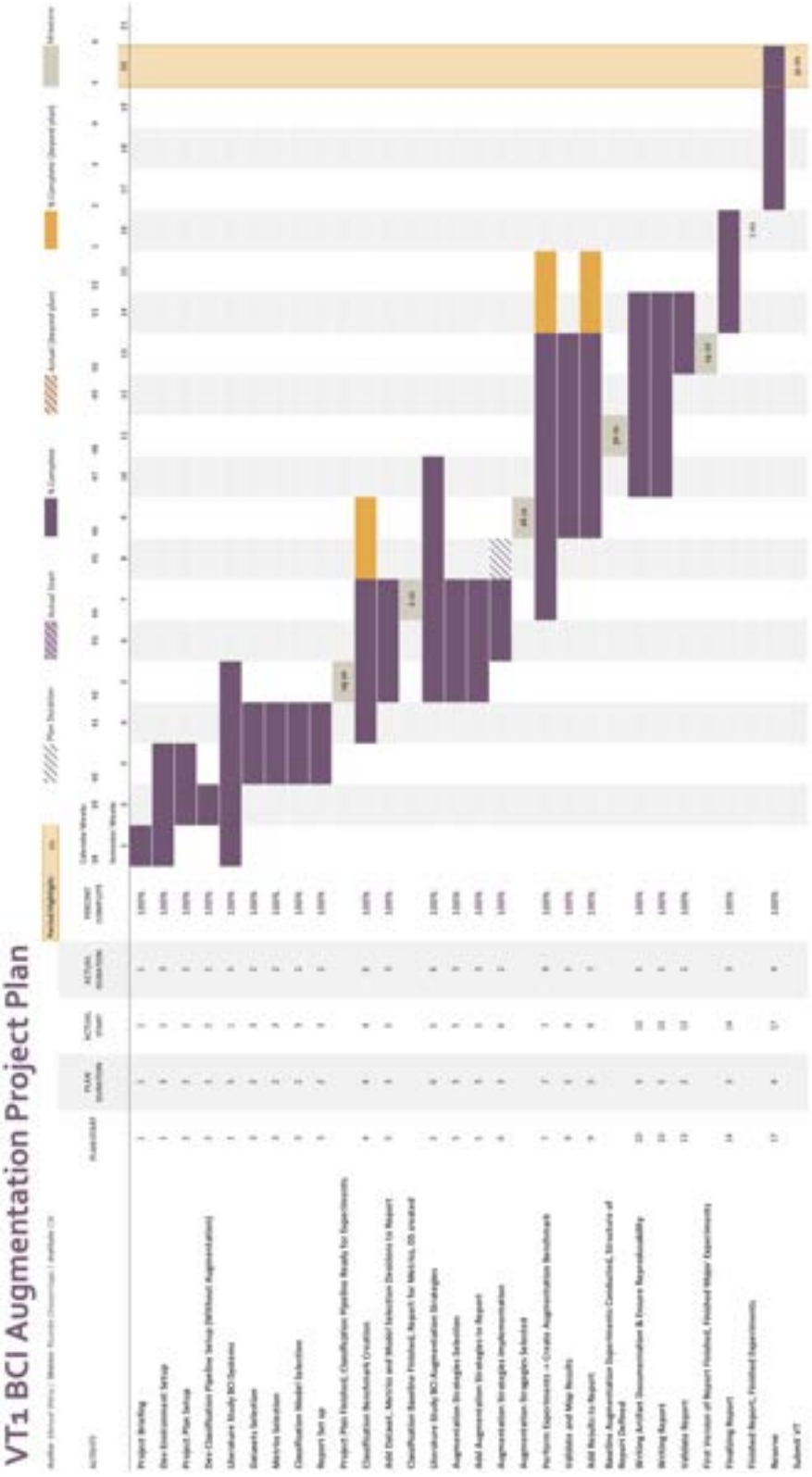


Figure A.10: Project Plan