

Bachelorarbeit, 7. Mai 2024

Überwachtes und selbstüberwachtes kontrastives maschinelles Lernen auf EEG- Daten zur Klassifikation mentaler Auslastung

Autorin: **Jeúsa Hamer**

Gutachter: **Dr. Felix Putze**
Prof. Dr. Udo Frese

ZUSAMMENFASSUNG

EEG als nicht invasive Messmethodik zur Überwachung von menschlichen Hirnaktivitäten eignet sich gut für die Nutzung in Brain Computer Interfaces. Hierbei bietet sich maschinelles Lernen und die Verwendung von neuronalen Netzen an, um die EEG-Daten für solche Anwendungen zu interpretieren. In diesem Bereich ist selbstüberwachtes Lernen eine Methodik, die gegenüber klassischem überwachten Lernen einige Vorteile verspricht und bei der Klassifikation von EEG-Daten teilweise auch schon erfolgreich erprobt wurde. Wenn es um die Klassifikation von mentaler Auslastung ging, fand selbstüberwachtes Lernen jedoch scheinbar noch keine Anwendung. In dieser Arbeit soll selbstüberwachtes Lernen für die Klassifikation mentaler Auslastung exploriert und mit überwachtem Lernen verglichen werden. Dafür wurden unter der Verwendung von Triplet-Loss überwacht und selbstüberwacht kontrastiv lernende Encoder trainiert, die Repräsentationen der EEG-Daten erlernen sollten. Bei der Architektur der Encoder handelte es sich um das EEGNet, ein kompaktes CNN. Die erlernten Repräsentationen wurden hinsichtlich ihrer Verwendbarkeit im Rahmen der Klassifikationsaufgabe evaluiert. Es wurden EEG-Daten einer Person aufgenommen, um zwei Datensätze für das Training der Modelle zu erstellen. Der gelabelte Datensatz umfasst vier Stufen mentaler Auslastung und wurde mithilfe eines eigens entwickelten Programms unter Verwendung der n -back-Aufgabe ($n \in \{0, 1, 2, 3\}$) erstellt. Für den zweiten Datensatz wurde eine größere Menge an ungelabelten EEG-Daten aufgenommen. Bei der Aufbereitung der Daten wurde die Verwendung von einem 0.1-45Hz-Bandpassfilter und einem 0.1-55Hz-Bandpassfilter miteinander verglichen, wobei eine stärkere Filterung mit dem 0.1-45Hz-Bandpassfilter als besser geeignet schien. Die überwacht trainierten Encoder, bei denen die EEG-Daten mit diesem Filter aufbereitet wurden, haben auf den Validierungsdaten solide Klassifikationsergebnisse mit einer Accuracy von 50% erzielen können. Auf einer separaten Testaufnahme war die Accuracy mit nur 31% jedoch deutlich geringer, da sich die Modelle stark auf die zum Training verwendeten EEG-Aufnahmen spezialisiert haben. Die selbstüberwacht trainierten Encoder haben sehr erfolgreich Repräsentationen der EEG-Daten erlernt. Diese ließen sich jedoch nicht gut für das anschließende Training eines Klassifizierers weiterverwenden. Die Accuracy für die Klassifikation der n -back-Daten war hier nicht besser als bei einer zufälligen Verteilung.

Inhaltsverzeichnis

1 Einleitung	1
2 Grundlagen	1
2.1 Elektroenzephalografie	1
2.2 Maschinelles Lernen	2
2.2.1 Überwachtes Lernen	2
2.2.2 Unüberwachtes Lernen	3
2.2.3 Selbstüberwachtes Lernen	4
2.3 Künstliche Neuronale Netze	4
2.3.1 Convolutional Neural Networks	5
3 Stand der Forschung	6
4 Daten	8
4.1 Datenaufnahme	8
4.2 N-back	9
4.2.1 Implementierung	9
4.2.2 Durchführung	9
4.3 Ungelabelte Daten	10
5 Methoden	10
5.1 Datenaufbereitung	10
5.2 EEGNet	13
5.3 Siamesisches Netzwerk	14
5.3.1 Triplet Loss	14
5.4 Klassifikation	15
5.4.1 K-nearest-neighbours	15
5.4.2 Klassifikationsnetz	16
5.5 Training	17
5.5.1 Überwacht	17
5.5.2 Selbstüberwachtt	18
5.5.2.1 Augmentationen	19
5.5.3 Klassifikationsnetz	20
6 Evaluation	21
6.1 Metriken	21
6.2 Ergebnisse	22
6.2.1 Überwacht	23
6.2.2 Selbstüberwachtt	28
6.2.3 Klassifikationsnetz	30
7 Diskussion	32
8 Fazit	36
Referenzen	38
Anhang	42

1 Einleitung

Messungen der Hirnaktivitäten beim Menschen mit EEG hatten ursprünglich vor allem medizinische Relevanz für die Diagnostik in der Neurologie. Sie lassen sich jedoch ebenso für die Entwicklung von sogenannten Brain Computer Interfaces (BCIs) nutzen [1], [2]. Solche Anwendungen erlauben die aktive oder passive Steuerung eines Geräts oder Programms alleine durch die Aktivität des Gehirns. BCIs haben ihrerseits auch Verwendungszwecke in der Medizin [3] und können beispielsweise Hilfsmittel für Menschen mit motorischen Einschränkungen darstellen. Weitere Anwendungsbereiche können darüber hinaus Smart Homes [4] oder Computer-Spiele [5] sein. BCIs zur Überwachung der mentalen Auslastung einer Person finden zum Beispiel bei adaptiven Lernsystemen [6] sowie bei der Überwachung des Müdigkeitszustands von Autofahrern [7] Anwendung. Auch können sie verwendet werden, um sicherzustellen, dass Piloten aufgrund von Überlastung nicht versehentlich Alarne missachten [8].

Für die Entwicklung solcher BCIs und der Verarbeitung der verwendeten Daten bietet sich maschinelles Lernen und die Nutzung von neuronalen Netzen an [9]. Ein Trend in diesem Bereich ist das sogenannte selbstüberwachte Lernen, das zunehmend auch bei der Nutzung von EEG-Daten Anwendung findet [10]. Selbstüberwachtes Lernen verspricht gegenüber klassischem überwachten Lernen, dass für das Erzielen guter Ergebnisse eine geringere Menge an gelabelten Trainingsdaten notwendig sei. Dies kann sehr hilfreich sein, da das Erstellen eines gelabelten EEG-Datensatzes oft mit großem Aufwand verbunden ist. Auch sollen auf diese Weise entwickelte Modelle über eine bessere Generalisierungsfähigkeit verfügen. Das Vorgehen beim selbstüberwachten Lernen besteht darin, dass auf ungelabelten Daten zunächst ein Modell entwickelt wird, das grundsätzliche Merkmale der Daten erlernt. Ein solches vortrainiertes Modell kann dann mit einem gelabelten Datensatz an die eigentliche Anwendungsaufgabe angepasst werden.

Während selbstüberwachtes Lernen auf EEG-Daten schon teilweise erfolgreich erprobt wurde [11]–[14], fand bei der Klassifikation unterschiedlicher Stufen mentaler Auslastung bisher in erster Linie klassisches überwachtes Lernen Anwendung [6]–[8], [15]–[19]. Diese Forschungsarbeit soll diese Lücke schließen. Dafür wurde ein kontrastiver selbstüberwachter Lernansatz zur Klassifikation mentaler Auslastung verfolgt und evaluiert, wobei zum Vergleich ebenso ein rein überwachtes Vorgehen auf eine möglichst ähnliche Weise umgesetzt wurde. Es wurden EEG-Messungen an einer Person durchgeführt, um zwei Datensätze zu erstellen. Zum einen wurde ein gelabelter Datensatz aufgenommen, wobei vier unterschiedliche Stufen mentaler Auslastung mithilfe der n -back-Aufgabe induziert wurden. Des Weiteren wurde ein ungelabelter Datensatz erfasst. Unter der Verwendung von Triplet-Loss wurde das EEGNet [20], ein Convolutional Neural Network, genutzt, um überwacht und selbstüberwacht lernende Encoder zu entwickeln, die sinnvolle Repräsentationen der EEG-Daten erlernen sollten. Diese wurden im Anschluss umfassend hinsichtlich ihrer Nutzung für die Klassifikation mentaler Auslastung im Rahmen der n -back-Aufgabe evaluiert.

2 Grundlagen

2.1 Elektroenzephalografie

Elektroenzephalografie (EEG) ist eine nicht invasive Messmethode, mit der sich die elektrischen Aktivitäten des Gehirns überwachen lassen. Hierbei werden sich verändernde Potentialfelder in der Hirnrinde gemessen. Diese entstehen durch die Bewegung von Ionen im Rahmen der chemischen synaptischen Erregungsübertragung [21].

Gemessen wird mit Elektroden, die an festen Positionen am Kopf angebracht werden. Die Positionen für die Elektrodenanbringung sind standardisiert. Jede Elektrode misst die Potentiale an der jeweilig angebrachten Stelle auf der Kopfoberfläche, wobei für die Spannungsabnahme außerdem Referenzelektroden notwendig sind.

EEG-Daten weisen schwingende Potentialmuster auf. Hier hat sich eine Aufteilung in Frequenzbänder etabliert, die empirisch entstanden, aber in der Regel auf physiologische Grundlagen zurückführbar sind. Mit EEG lassen sich an der Kopfhaut Potentialschwankungen mit einer Frequenz von bis zu 30Hz messen [21]. Die Beta-Wellen (13-30Hz) beschreiben die am schnellsten schwingenden EEG-Signale. Das Alpha-Band (8-13Hz) charakterisiert die messbare Grundaktivität im Ruhezustand. Langsamere Frequenzen unterhalb des Alpha-Bandes werden durch das Theta-Band (4-8Hz) sowie das Delta-Band (0.5-4Hz) beschrieben.

EEG-Messungen werden in der medizinischen Diagnostik genutzt, um Auffälligkeiten in der Hirnfunktionalität festzustellen, beispielsweise für die Diagnose von Epilepsie. Auch zur Überwachung des Schlafes ist EEG anwendbar. EEG-Daten lassen sich jedoch ebenso für die Entwicklung von Brain-Computer-Interfaces nutzen, wobei Änderungen in den Potentialmustern interpretiert werden, um Computer-Anwendungen zu steuern.

2.2 Maschinelles Lernen

Maschinelles Lernen [22], [23] unterscheidet sich in der Methodik grundsätzlich von der Entwicklung klassischer Programme. Bei solchen ist durch den Programmiercode eindeutig vorgeschrieben, wann das Programm sich wie verhalten soll. Beim maschinellen Lernen hingegen wird ein Modell auf Basis von einem Datensatz entwickelt, und das Modell lernt auf Grundlage dieser Daten selber, welche Ausgabe in welchem Fall angebracht ist. In dieser Arbeit sollten solche Modelle lernen, eine Klassifikationsaufgabe zu lösen. Aufgrund der Merkmale eines einzelnen Datenpunktes, auch *Sample* genannt, soll bei einer Problemstellung dieser Art entschieden werden, in welche Klasse dieses Sample eingesortiert werden soll. Die Klasse eines Samples wird auch als sein *Label* bezeichnet. Solche Modelle werden oft, wie auch in dieser Arbeit, durch *künstliche neuronale Netze* dargestellt (siehe Abschnitt 2.3).

Ein Modell wird auf einem Datensatz trainiert, um Muster in den Daten im Sinne der jeweiligen Anwendungsaufgabe erkennen zu lernen. In dieser Arbeit stellte die Anwendungsaufgabe die Klassifikation von Stufen mentaler Auslastung dar. Für die Evaluation eines trainierten Modells wird in der Regel ein separater Datensatz verwendet, der beim Training nicht genutzt wurde und dessen Daten das Modell daher noch nicht kennengelernt hat. Ein solcher Testdatensatz ist notwendig, da es passieren kann, dass ein Modell auf den Trainingsdaten sehr gute Ergebnisse erzielt, das erlernte Wissen aber nicht gut auf unbekannte Daten anwenden kann. Dann hat eine *Überanpassung (Overfitting)* auf die Trainingsdaten stattgefunden, wodurch das Modell nicht auf fremde Daten generalisieren kann. Ein solches Modell hat lediglich die zum Training verwendeten Daten auswendig gelernt, statt die für die Klassifikation relevanten Strukturen in den Daten zu identifizieren. Neben Trainings- und Testdaten gibt es in der Regel noch weitere Daten, die für die Validierung verwendet werden. Die Validierungsdaten werden wie die Testdaten nicht zum Training eines Modells verwendet. Sie dienen aber bei der Entwicklung der Modelle der Überwachung der Zwischenergebnisse. Um mit einem Modell bestmögliche Ergebnisse zu erzielen, ist es oft notwendig, unterschiedliche Einstellungsmöglichkeiten für das Training auszuprobieren, man spricht hier von einem *Tuning der Hyperparameter*. Hier werden die Ergebnisse auf den Validierungsdaten genutzt, um ein Modell und die verwendeten Hyperparameter zu überprüfen. Bei diesem Vorgehen kann wiederum auch eine gewisse Anpassung des Modells an den Validierungsdatensatz stattfinden. Daher sind die separaten Testdaten notwendig, die erst zum Schluss für die Evaluation eines fertig entwickelten Modells genutzt werden.

2.2.1 Überwachtes Lernen

Überwachtes Lernen ist eine Art des maschinellen Lernens [22], [23]. Hierbei wird ein Trainingsdatensatz verwendet, bei dem die Labels, also bei einem Klassifikationsproblem die Klassenzugehörigkeiten der Datenpunkte, bekannt sind. Solche gelabelten Datensätze zu erstellen ist oft sehr aufwendig. Sie entstehen teilweise mithilfe von menschlichen Experten, die den Samples Labels zuordnen. In dieser

Arbeit wurde ein Programm geschrieben, mithilfe dessen sich ein gelabelter EEG-Datensatz erstellen lässt, bei dem sich die Samples unterschiedlichen Stufen mentaler Auslastung zuordnen lassen.

Modelle, die überwacht trainiert werden, sollen anhand des gelabelten Trainingsdatensatzes lernen, Samples auf die gleiche Weise Klassen zuzuordnen, wie es für die Samples in dem Datensatz erfolgt ist. Dafür müssen sie die für die Klassifikation wichtigen Merkmale der Daten erkennen lernen. Für neuronale Netze und viele andere Modelle geschieht dies iterativ. Ein Modell versucht für eine Reihe an Samples jeweils die Klasse vorherzusagen. Die vorhergesagten Labels werden dann mit den wahren Labels, der sogenannten *Ground Truth*, verglichen, und es wird mit einer *Loss-Funktion* die Abweichung zwischen der Vorhersage des Modells und den tatsächlichen Labels berechnet. Im Laufe des Trainings versucht ein Modell mit jeder Iteration den Loss, also die Abweichung von der Ground Truth, zu verringern und so die Klassifikation zu verbessern.

2.2.2 Unüberwachtes Lernen

Beim *unüberwachten Lernen* wird kein gelabelter Datensatz benötigt [22], [23]. Diese Methodik kommt auch mit Daten ohne Labels aus und hat in der Regel andere Anwendungszwecke als das überwachte Lernen.

Eine Variante des unüberwachten Lernens ist das *Clustering*. Hierbei versucht ein Modell aufgrund der Merkmale der Samples Gruppierungen in den Daten zu erkennen, die sogenannten *Cluster*. Innerhalb eines Clusters befinden sich dann Datenpunkte mit ähnlichen Eigenschaften. Inwieweit die dabei erstellten Cluster auch sinnvolle Klassen für diese Daten darstellen ist dann in der Regel noch unklar und muss weiter untersucht werden.

Unüberwachtes Lernen kann auch genutzt werden, um *Repräsentationen* der Daten zu erlernen, welche die relevanten Merkmale eines Samples gut zusammenfassen. *Autoencoder* sind ein Beispiel für eine Art von neuronalen Netzen, die solche komprimierten Repräsentationen erlernen sollen. Wie in Abbildung 1 abgebildet, bestehen diese Modelle aus zwei Teilen, einem *Encoder* und einem *Decoder*. Autoencoder lernen anhand einer Eingabe, diese als niedrigdimensionale Repräsentation zu kodieren, um daraus eine Ausgabe zu generieren, die der ursprünglichen Eingabe möglichst ähnlich ist. Der Encoder reduziert dabei die Dimensionalität der Eingabe. Ein 512x512 Pixel großes Bild wird zum Beispiel reduziert auf eine Größe von 64x64 Pixel. Der Decoder soll dann aus dieser Repräsentation die ursprüngliche Eingabe rekonstruieren. Ein Autoencoder lernt so, die wichtigen Merkmale eines Samples in einer kompakten Repräsentation darzustellen. Diese Repräsentationen lassen sich dann für andere Zwecke weiterverwenden.

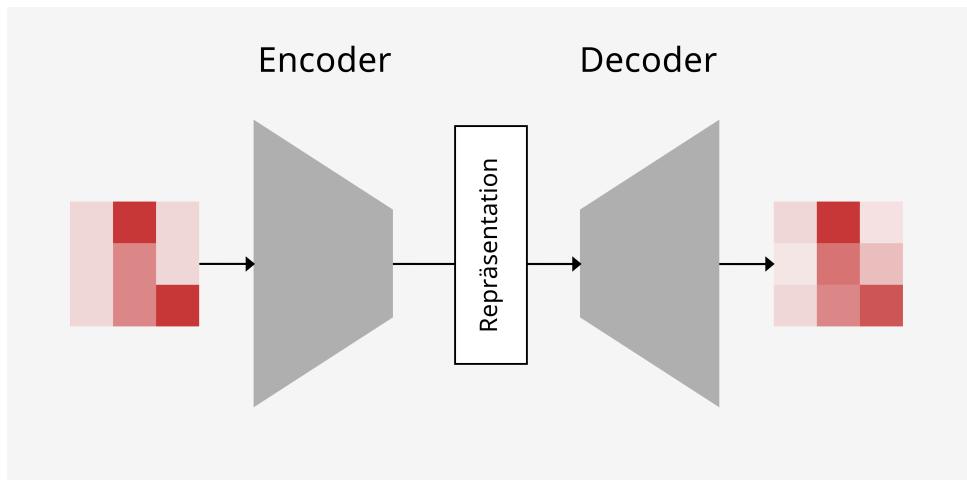


Abbildung 1: Autoencoder

2.2.3 Selbstüberwachtes Lernen

Das *selbstüberwachte Lernen* ist eine neuere Methode des maschinellen Lernens, die nicht eindeutig definiert ist, sich aber zwischen überwachtem und unüberwachtem Lernen bewegt [24]. Nach Rafiei et al. [10] lässt sich selbstüberwachtes Lernen in drei Teilaufgaben zerlegen.

Den ersten Teil stellt die *Pretext-Aufgabe* dar, die sich im Bereich des unüberwachten Lernens befindet. Mit dieser Aufgabe wird zum Beispiel ein Modell entwickelt, das in der Lage ist, sinnvolle Repräsentationen von den verwendeten Daten zu erstellen. Diese sind noch nicht an eine spezifische Aufgabe angepasst, sondern enthalten generelle Merkmale der Daten. Für die Entwicklung des Pretext-Modells können ungelabelte Daten verwendet werden, dieser Datensatz sollte jedoch möglichst groß sein. Im zweiten Schritt findet eine Anpassung dieses Modells statt, um es für den eigentlichen Anwendungszweck, die *Downstream-Aufgabe*, verwendbar zu machen. Die dritte Aufgabe stellt dann das Training im Rahmen der Downstream-Aufgabe selber dar. Hierbei wird eine vergleichsweise kleine Menge gelabelter Daten verwendet. In diesem Schritt können zum Beispiel die vom Pretext-Modell erlernten Repräsentationen für das überwachte Training eines Klassifizierers weiterverwendet werden.

Die Pretext-Aufgabe kann generativ oder kontrastiv sein [24], [10]. Ein generativer Ansatz wäre zum Beispiel die Verwendung eines Autoencoders, um Repräsentationen der Daten zu erstellen. In dieser Arbeit wurde ein kontrastiver Ansatz verfolgt. Hierbei lernt ein Modell, indem es mehrere Eingaben hinsichtlich ihrer Gemeinsamkeiten und Unterschiede vergleicht.

Von der Verwendung selbstüberwachten Lernens verspricht man sich gegenüber klassischem überwachten Lernen, dass auf diese Weise entwickelte Modelle über bessere Generalisierungsfähigkeiten auf dem Modell unbekannten Daten verfügen. Auch sollen so für das überwachte Training im Rahmen der Downstream-Aufgabe kleinere Mengen an gelabelten Trainingsdaten notwendig sein, deren Erstellung oft mit großem Aufwand verbunden ist. Dafür wird in der Regel eine recht große Menge an ungelabelten Daten für die Entwicklung der Modelle benötigt, anhand derer im Rahmen der Pretext-Aufgabe die grundsätzlichen Merkmale der Daten erlernt werden.

2.3 Künstliche Neuronale Netze

Künstliche neuronale Netze sind durch Computer simulierte Systeme der Informationsverarbeitung, die grob an der Funktionsweise von Gehirnen orientiert sind.

Die wohl einfachste Form solcher neuronaler Netze sind die *mehrschichtigen Perzeptren (multilayered perceptrons, MLPs)* [25]. In Abbildung 2 ist die Funktionsweise eines einzelnen Perzeptrons dargestellt. Ein solches künstliche Neuron bekommt einen Eingabevektor, dessen Komponenten als gewichtete Summe zusammengefasst werden. Zusätzlich zu den Gewichten enthält ein Perzepron auch noch eine

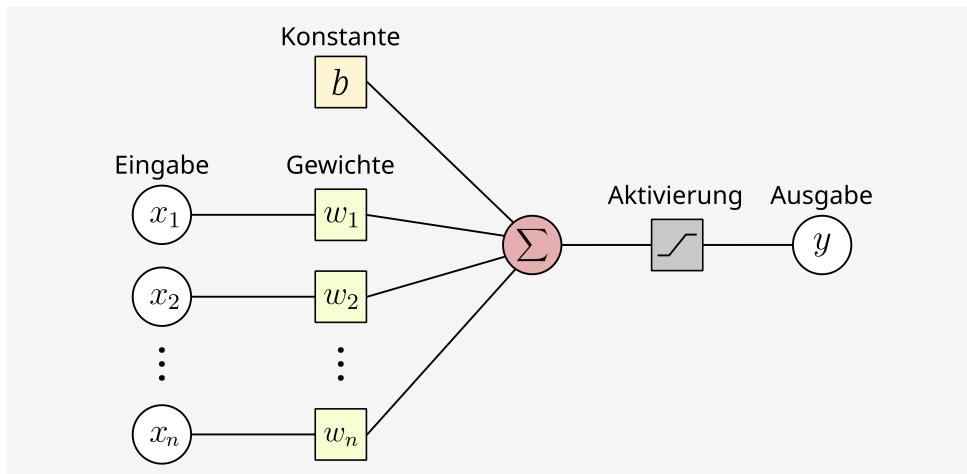


Abbildung 2: Funktionsweise eines Perzeptrons

Konstante, die ebenso in die Summe eingeht. Die Gewichte und die Konstante stellen die variablen lernbaren Parameter des Neurons dar. Nach der Summe folgt eine Aktivierungsfunktion, um zum Beispiel alle darin eingehenden Werte auf einen Wertebereich von $[0, 1]$ abzubilden.

Diese Neuronen lassen sich nun verknüpfen zu einem MLP, einem einfachen neuronalen Netz, dargestellt in Abbildung 3. Hier sind die Neuronen immer in Schichten angeordnet, wobei die aufeinanderfolgenden Schichten *vollständig verknüpft* (*fully connected, FC*) sind. Jede Ausgabe eines Neurons aus der vorherigen Schicht ist Eingabe für jedes Neuron der nächsten Schicht. Mit solchen simplen Netzen lassen sich schon einfache Aufgabenstellungen lösen. Beim Training eines neuronalen Netzes werden seine Parameter, bei einem MLP also die Gewichte und Konstanten jedes Neurons, so optimiert, dass die Vorhersagen auf dem Trainingsdatensatz möglichst gut funktionieren und der Loss, also die Abweichung vom erwünschten Ergebnis, so klein wie möglich wird.

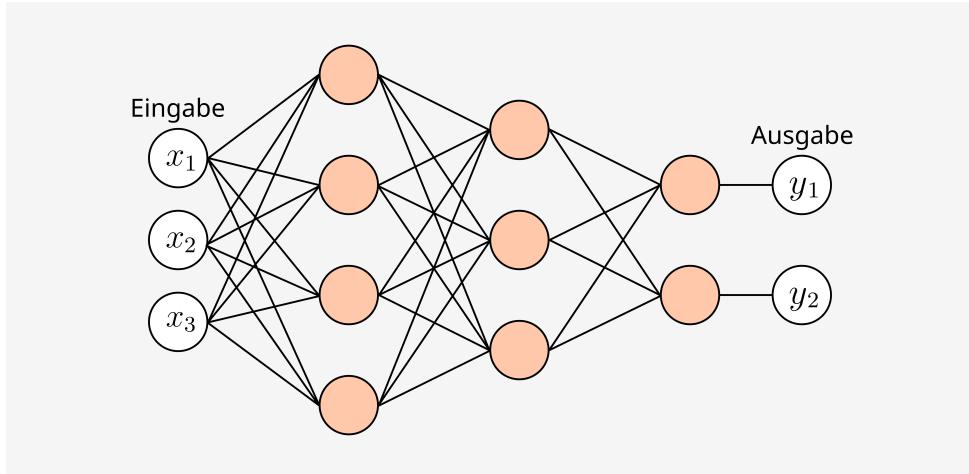


Abbildung 3: Mehrschichtiges Perzeptron

2.3.1 Convolutional Neural Networks

Neben den einfachen vollständig verknüpften Schichten werden noch andere spezialisiertere Schichtarten in neuronalen Netzen verwendet. Dazu gehören *Convolutional Layer*, die ursprünglich aus der Bildverarbeitung stammen, sich aber auch bei anderen Datentypen und so auch bei EEG-Daten anwenden lassen. Ein Netz mit solchen Schichten wird ein *Convolutional Neural Network (CNN)*, genannt [23].

Eine Convolution oder Faltung ist eine lineare Operation, bei der sich ein *Filter* in *Schritten* über eine Eingabe bewegt. Filter- und Schrittgröße einer Schicht sind dabei festlegbar. In dem Beispiel in Abbildung 4 beträgt die Filtergröße $(3, 3)$, wobei der Filter sich in $(1, 1)$ Schritten, also über jeden einzelnen Pixel des Eingabebildes, bewegt. Ein Wert der Ausgabe ergibt sich als Skalarprodukt zwischen Filter und der jeweiligen Eingabematrix. Damit die äußeren Pixel der Eingabe in gleichem Maße in die Ausgabe einfließen wie die im Inneren liegenden, muss um das Eingabebild herum ein *Padding* erfolgen. Dadurch wird die Eingabe künstlich vergrößert. In dem Beispiel wird das Padding erstellt, indem die äußeren Pixel des Originalbildes dupliziert werden. Auf solche Convolutional Layer folgen im Anschluss in der Regel *Pooling-Layer*. Diese fassen benachbarte Werte zusammen, wodurch die Dimensionalität reduziert, das rezeptive Feld vergrößert und Invarianz für kleine Translationen gefördert wird [23].

Die erlernbaren Parameter solcher Convolutional Layer sind die Filter. In den ersten Schichten eines CNNs können die Filter einfache Strukturen in den Eingabedaten erkennen. Bei Bildern als Eingabe könnte ein Filter hier beispielsweise erlernen, senkrechte Kanten zu identifizieren. Ein anderer Filter könnte horizontale Kanten erkennen lernen. In den darauffolgenden Schichten werden die vorherig

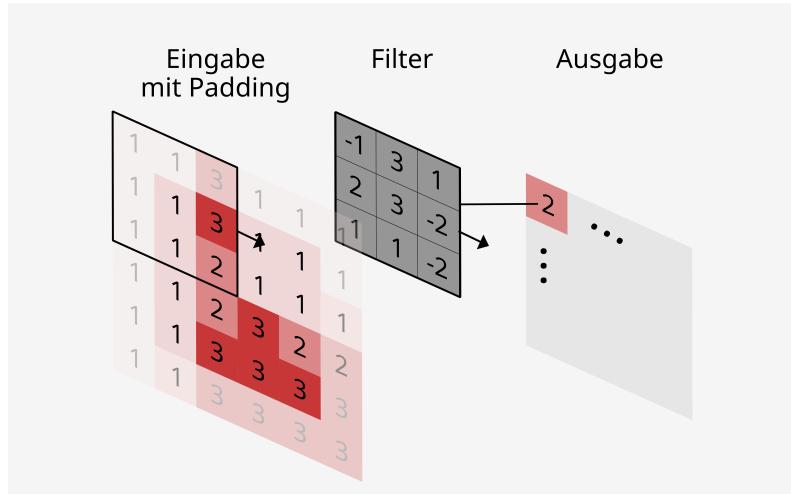


Abbildung 4: 2D-Convolution

erlernten Informationen zu abstrakteren und komplexeren Konzepten verknüpft. So können in diesen tieferen Schichten bei einem CNN, das für die Bildverarbeitung verwendet wird, zum Beispiel die vorherigen Kanteninformationen genutzt werden, um konkrete Gegenstände in einem Bild zu identifizieren.

3 Stand der Forschung

Kontrastives selbstüberwachtes Lernen findet zunehmend Anklang, wenn es um die Klassifikation von EEG-Daten [10] und anderen zeitlichen Biosignalen geht [26], [27].

Xu et al. [11] entwickelten beispielsweise ein auf EEG-Daten selbstüberwacht lernendes CNN für die Epilepsieerkennung. Die Pretext-Aufgabe bestand hier aus der Klassifikation unterschiedlicher zeitlicher Transformationen der Daten. Auf Basis dieses Modells wurde dann ein Anomaly-Detektions-System für die Unterscheidung zwischen normalen und abnormalen EEG-Daten entwickelt, das bessere Ergebnisse erzielte als vergleichbare Modelle.

Auch Banville et al. [13] haben ein CNN als Pretext-Modell verwendet. Hier wurden Encoder in Form von siamesischen Netzwerken entwickelt, wobei zwei temporale Aufgaben gestellt wurden. Einer der selbstüberwacht entwickelten Encoder sollte lernen, ob zwei eingegebene Samples zeitlich nahe beianander oder weit voneinander entfernt sind. Ein weiterer Encoder hat drei Samples bekommen, wobei er identifizieren sollte, ob sich diese in der zeitlich richtigen Abfolge befinden oder ob sie durchmischt wurden. Evaluiert wurden die Encoder auf EEG-Datensätzen für die Klassifikation von Schlafphasen. Die Repräsentationen der Encoder wurden für die Downstream-Aufgabe genutzt, um Modelle der multinomialen logistischen Regression zu trainieren. Im Vergleich zu einem rein überwachten Ansatz wurden hier vor allem bessere Ergebnisse mit den selbstüberwacht vortrainierten Encodern erzielt, wenn nur ein kleiner Anteil an gelabelten Daten verwendet wurde. Bei einer größeren Menge an gelabelten Daten schwand der Vorsprung der selbstüberwachten Modelle.

Mohsenvand et al. [14] wandelten SimCLR [28], einen kontrastiven selbstüberwachten Ansatz aus der Bildverarbeitung, ab für die Verwendung mit EEG-Daten. Das Pretext-Modell bestand hier aus einem Encoder und einem Projektor, wobei die Eingabe für einen Encoder nur ein einzelner EEG-Kanal war. Dies erlaubte es den Autoren, einfach mehrere EEG-Datensätze mit unterschiedlicher Kanalanzahl für das selbstüberwachte Training zu kombinieren, was der Bewältigung der Downstream-Aufgabe zuträglich war. Für die Pretext-Aufgabe wird hier ein Sample auf zwei unterschiedliche Weisen augmentiert. Mithilfe einer kontrastiven Loss-Funktion sollen die beiden Signale im Ausgaberaum des Projektors möglichst ähnlich angeordnet werden. Die Klassifikation im Rahmen der Downstream-Aufgabe

fand dann für Emotionen, Schlafphasen und normale/abnormale EEG-Daten statt. Hierfür wurde ein neuer Projektor trainiert. Auf zwei der drei Datensätze wurden im Vergleich zu vorherigen überwachten Ansätzen mit den selbstüberwacht entwickelten Modellen bessere Ergebnisse erzielt. Außerdem wurde in dieser Studie festgestellt, dass eine Reihe an unterschiedlichen Augmentationen beim selbstüberwachten Training zu besseren Resultaten als die Verwendung von nur einzelnen Augmentationen führt. Darüber hinaus haben sie die Datenmenge künstlich vergrößert, indem sie bestehende EEG-Kanäle zu neuen Kanälen rekombiniert haben. Auch dies schien förderlich zu sein für die Klassifikation.

Yang et al. [12] nutzten ebenfalls augmentierte EEG-Samples. Darüber hinaus wurde eine Repräsentation der „Welt“, also des ganzen Datensatzes, erstellt. Die von einem selbstüberwacht trainierten Convolutional Encoder ausgegebenen Repräsentationen der augmentierten Daten sollten dabei möglichst ähnlich zueinander und unterschiedlich von der Welt-Repräsentation sein. Die erlernten Repräsentationen wurden anschließend für die Entwicklung eines überwacht lernenden linearen Klassifizierers verwendet. Bei der Klassifikation von Schlafphasen konnte dieser Ansatz etwas bessere Ergebnisse als andere selbstüberwachte Methodiken und ein zum Vergleich entwickeltes überwacht lernendes Modell erzielen.

Wenn es um die Klassifikation mentaler Auslastung geht, ob in Bezug auf die *n*-back-Aufgabe oder nicht, so wurde hier die Verwendung von selbstüberwachtem Lernen scheinbar noch nicht getestet. Hier ließen sich lediglich Studien finden, in denen klassisch überwacht lernende Modelle entwickelt wurden.

Herff et al. [18] entwickelten probandenspezifische Klassifizierer für vier Stufen von mentaler Auslastung. Hier handelte es sich um Daten, die mit der *n*-back-Aufgabe induziert wurden. Die Aufnahmetethodik war funktionelle Nahinfrarotspektroskopie (fNRIS), ein Verfahren, das am Kopf Änderungen der Blutsauerstoffsättigung misst, die durch neuronale Aktivitäten hervorgerufen werden. Mithilfe von linearer Diskriminanzanalyse wurden Vier-Klassen-Klassifizierer entwickelt, die durchschnittlich eine Accuracy von 45% erzielten. Diese Studie testete außerdem unterschiedliche Fenstergrößen für die Sampleerstellung. Samples, deren Länge 25s Datenaufnahme entsprachen, haben hier die besten Ergebnisse erzielt.

Saadati et al. [16] haben MLPs auf EEG- und fNRIS-Daten für ein *n*-back-Problem mit vier Klassen trainiert. Die EEG-Samples hatten Längen zwischen 2s und 5s und wurden im Vorhinein mittels ereigniskorrelierter Desynchronisation/Synchronisation aufbereitet. Auf den EEG-Daten alleine entwickelte Klassifizierer, die pro Person erstellt wurden, erreichten durchschnittlich eine Accuracy von 67%. Noch bessere Klassifikationsergebnisse wurden durch die Kombination von EEG- und fNRIS-Daten erzielt.

Auch Liu et al. [19] haben sowohl EEG- als auch fNRIS-Daten für eine Klassifikation mentaler Auslastung pro Proband unter der Nutzung von *n*-back verwendet. Die EEG-Samples wurden mit Fenstergrößen von 2.5s Länge erstellt und wurden auch hier aufbereitet, bevor sie als Eingabe für einen Klassifizierer dienten. Für jedes Sample wurde das Leistungsdichtespektrum erstellt, womit dann lineare Diskriminanzanalyse für die Klassifikation durchgeführt wurde. Hier hat die Kombination von EEG und fNRIS ebenso am besten funktioniert, mit den EEG-Daten alleine wurden bei einem Klassifikationsproblem mit drei Klassen durchschnittlich Accuracies von 43% erreicht.

Die bisherige Forschung bezüglich der Klassifikation von mentaler Auslastung unter der Verwendung von Daten, die die Aktivität des Gehirns messen, nutzt scheinbar noch kein selbstüberwachtes Lernen. Auch findet hier häufig eine recht aufwendige Aufbereitung der EEG-Daten statt, oder sie werden mit anderen Messungen kombiniert. In dieser Arbeit wurde nun selbstüberwachtes Lernen für die Klassifikation mentaler Auslastung erprobt. Dabei sollten die EEG-Daten ohne ausführliche Bereinigung oder Aufbereitung verwendet werden, um in der Pretext-Task Encoder zu trainieren, die sinnvolle Re-

präsentationen der Daten erlernen sollten. Wie bei vielen anderen der geschilderten selbstüberwachten Ansätze kam dabei ein CNN, das EEGNet [20], zum Einsatz. Die Methodik des selbstüberwachten Lernens und die dafür verwendeten Augmentationen waren angelehnt an das Vorgehen von Mohsenvand et al. [14], da es auf den dort verwendeten EEG-Datensätzen zu guten Ergebnissen geführt hat. Im Gegensatz zu anderer Forschung dieses Bereichs wurden für diese Arbeit außerdem mehrere EEG-Aufnahmen für eine Person erstellt, während in vielen anderen Studien nur eine Aufnahme pro Proband existiert. Dies erlaubt hier eine ausführlichere Evaluationsmethodik der entwickelten Modelle.

4 Daten

Im Rahmen dieser Arbeit wurden zwei EEG-Datensätze erstellt. Einerseits wurden gelabelte Daten aufgenommen, bei denen sich die EEG-Daten Klassen zuordnen lassen. Diese Klassen entsprechen Stufen unterschiedlicher mentaler Auslastung, die mit der *n-back*-Aufgabe induziert wurden (siehe Abschnitt 4.2). Außerdem wurden ungelabelte EEG-Daten ohne Klassenzugehörigkeit aufgenommen (siehe Abschnitt 4.3). Alle für diese Arbeit aufgenommenen EEG-Daten stammen lediglich von mir selbst, da eine Studie mit weiteren Versuchspersonen den Rahmen dieser Abschlussarbeit gesprengt hätte. In diesem Forschungsbereich ist es jedoch nicht unüblich, dass Klassifizierer spezifisch für einzelne Personen entwickelt werden, da sich die EEG-Daten zwischen unterschiedlichen Personen relativ stark unterscheiden können, was die Entwicklung von Modellen, die über mehrere Personen hinweg generalisieren können, schwierig macht [29].

4.1 Datenaufnahme

Die Aufnahme der EEG-Daten erfolgte mit dem *Unicorn Hybrid Black*¹ Headset, das die Daten in acht Kanälen und mit einer Frequenz von 250Hz misst. In Abbildung 5 ist die Haube, auf der die acht Messelektroden angebracht sind, sowie die Positionen der zwei Referenzelektroden L und R dargestellt. Nach dem 10-20-System, dem international gängigen und standardisierten System für Elektrodenplatzierungen, sind die Positionen der acht Elektroden Fz, C3, Cz, C4, Pz, PO7, Oz und PO8. Die Elektroden lassen sich trocken so wie sie sind oder feucht zusammen mit einem Gel verwenden, was zu einer besseren Signalqualität führt, dafür aber einen höheren Präparationsaufwand mit sich bringt. Es wurde sich hier für Trockenelektroden entschieden, um die Aufnahme der Daten möglichst niedrigschwellig zu halten. Mit der zum Headset zugehörigen Unicorn-Software lässt sich eine kabellose Verbindung mit dem Headset herstellen, und die gemessenen EEG-Daten können abgespeichert werden. Das

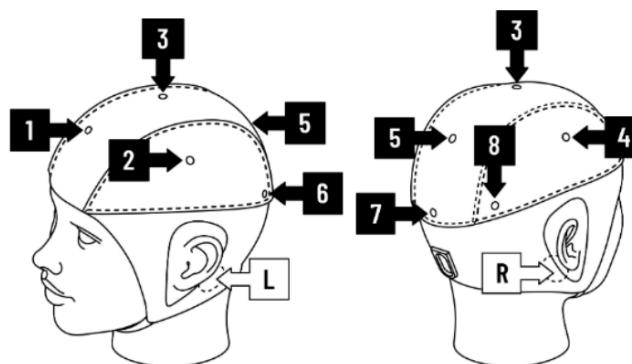


Abbildung 5:

Unicorn Hybrid Black Elektroden-Haube

Quelle: „User Manual for Unicorn Brain Interface Hybrid Black“, V1.18.00, 2021, S. 12,
g.tec neurotechnology GmbH Austria

¹g.tec medical engineering GmbH, <https://www.unicorn-bi.com/>

Programm, das zum Aufnehmen der ungelabelten Daten verwendet wurde, war der *Unicorn Recorder*¹². Um gelabelte EEG-Daten zusammen mit der *n-back*-Anwendung zu erzeugen, wurde das *Unicorn LSL Interface*¹² verwendet, das eine Schnittstelle zu *Lab Streaming Layer (LSL)* [30] Anwendungen herstellt. Hierbei werden die EEG-Daten als LSL-Datenstrom weitergeleitet und können mit anderen Datenströmen synchronisiert werden, für weitere Informationen siehe folgenden Abschnitt 4.2.1. Die Daten wurden dann mit dem *LabRecorder* [31] von LSL [30] aufgenommen. Alle aufgenommenen Daten waren zunächst unbearbeitet und ungefiltert. Das Headset misst auch Daten über einen integrierten Beschleunigungs- und einen Gyrosensor, welche in dieser Arbeit aber keine Verwendung finden.

4.2 N-back

Die *n-back*-Aufgabe ist ein am Computer durchgeführter Dauerbelastungstest, wobei n hier einen Wert aus $\{0, 1, 2, 3\}$ annehmen kann. Je höher der Wert von n , desto schwieriger ist die gestellte Aufgabe und desto höher ist die beim Probanden induzierte mentale Auslastung. Die vier möglichen Werte für n korrespondieren also mit vier möglichen Klassen, die wiederum für unterschiedliche Stufen mentaler Auslastung stehen.

Die Aufgabe läuft wie folgt ab: Auf einem Bildschirm wird eine Sequenz an Buchstaben dargestellt, wobei immer ein Buchstabe zur Zeit abgebildet ist. Der Nutzer soll für jeden angezeigten Buchstaben angeben, ob es sich dabei um den gleichen Buchstaben handelt, der n Schritte vorher angezeigt wurde. Bei 2-back könnte eine solche Sequenz beispielsweise so aussehen:

c d p w **p** d w **d** w k k

Dabei sind die fett gedruckten Buchstaben hier diejenigen, die n Schritte vorher ebenso angezeigt wurden, worauf der Proband durch eine Tasteneingabe entsprechend reagieren soll. 0-back ist als die einfachste Aufgabenstellung extra definiert. Hierbei muss vom Nutzer lediglich bestätigt werden, ob es sich bei dem aktuelle Buchstaben um ein „x“ handelt oder nicht.

4.2.1 Implementierung

Die Logik der *n-back*-Anwendung wurde in Python implementiert, wobei die Verwendung der Bibliothek *Eel* [32] eine Erstellung der Benutzeroberfläche mit HTML und JavaScript erlaubte. Die Anwendung läuft dann im Browser. Um nun automatisiert gelabelte EEG-Daten zu erhalten, wurde, wie bereits erwähnt, Lab Streaming Layer [30] verwendet. Ein LSL-Datenstrom besteht dann aus den EEG-Daten, die bei der Testperson während des Bearbeitens der *n-back*-Aufgabe gemessen werden. Aus der *n-back*-Anwendung selber kommt gleichzeitig ein weiterer Datenstrom, der durch Verwendung der Bibliothek *PyLSL* [33] entsprechende Marker über den aktuellen Status der Aufgabe sendet. Der *LabRecorder* [31] von LSL erlaubt dann das synchronisierte Abspeichern der beiden Datenströme. Startet nun zum Beispiel ein 1-back-Block in der Anwendung, den der Nutzer bewältigen soll, sendet die Anwendung die beiden Marker „start_block“ gefolgt von „n=1“. Endet der Block, folgt ein Marker „end_block“. So lassen sich einfach alle EEG-Daten, die zwischen „start_block“ und „end_block“ aufgenommen wurden, der Klasse $n = 1$ zuordnen.

4.2.2 Durchführung

Die Sequenz an angezeigten Symbolen besteht in der für diese Arbeit erstellten Version der *n-back*-Aufgabe nur aus Konsonanten. Vokale werden nicht verwendet, um Lautbildung, die das Merken der Buchstabenabfolgen erleichtern könnte, zu verhindern [17]. Demnach stehen 21 unterschiedliche Zeichen zur Verfügung. In der entwickelten Anwendung lässt sich eine Sitzung starten, die aus acht *n-back*-Blöcken besteht. Für einen Block nimmt n einen zufälligen Wert aus $\{0, 1, 2, 3\}$ an, wobei jede der vier möglichen Blockvarianten in einer Sitzung jeweils zwei Mal vorkommt. Jede Variante von

¹²v1.18.00

n -back kommt hierbei jeweils ein Mal in der ersten und in der zweiten Hälfte der Sitzung vor, wobei ausgeschlossen wird, dass zwei Blöcke mit demselben Wert für n aufeinander folgen. Ein Zeichen wird für 1s angezeigt und ist gefolgt von 2s Pause bis zur Anzeige des nächsten Zeichens. Insgesamt besteht ein Block aus 30 Zeichen bei einer Gesamtlänge von 90s. Bei einem Drittel der Zeichen trifft die n -back-Bedingung zu, das heißt, dass n Schritte vorher dasselbe Zeichen oder „x“ im Falle von 0-back angezeigt wurde. In diesem Fall soll der Nutzer die Taste „j“ drücken. Trifft die Bedingung nicht zu, soll „f“ gedrückt werden. Über eine Farbänderung des Hintergrunds bekommt der Proband visuelle Rückmeldung darüber, ob die Eingabe korrekt (grün) oder falsch (rot) war. Zwischen zwei Blöcken gibt es eine Unterbrechung von 15s, wovon 10s eine Pause bilden, gefolgt von 5s, in denen die Information angezeigt wird, welche n -back-Variante im nächsten Block folgt. Die Gesamtlänge einer Sitzung beträgt 14 Minuten, wovon 2 Minuten Pausen zwischen den Blöcken sind.

Für diese Arbeit wurden vier n -back-Sitzungen aufgenommen. Dies geschah an unterschiedlichen Tagen. Der resultierende n -back-Datensatz umfasst somit insgesamt 48 Minuten gelabelte EEG-Daten.

4.3 Ungelabelte Daten

Für das Training von Modellen, die selbstüberwacht lernen, war neben den gelabelten auch eine größere Menge weiterer EEG-Daten nötig. Diese wurden in insgesamt neun Sitzungen aufgenommen, in denen ich am Schreibtisch stand oder saß und am Computer gearbeitet habe, wobei ich währenddessen das EEG-Headset aufhatte. In zwei der Sitzungen habe ich teilweise auch Klavier gespielt, in einer habe ich an einem Online-Meeting teilgenommen. Die Aufnahmen wurden an unterschiedlichen Tagen gemacht. Der ungelabelte Datensatz umfasst insgesamt 10.3 Stunden an ungelabelten EEG-Daten.

5 Methoden

5.1 Datenaufbereitung

Bevor die EEG-Daten für das Training neuronaler Netze verwendet werden konnten, mussten sie aufbereitet werden. Hierfür wurden unter anderem die Bibliotheken *pandas*³ [34] und *MNE* [35] verwendet. EEG-Daten sind anfällig für Artefakte, die durch Umwelteinflüsse kommen oder physiologische Ursachen haben können. Ein mögliches Artefakt kann durch elektrische Ströme in Geräten oder Steckdosen in der Umgebung ausgelöst werden [36]. In Abbildung 6 ist erkennbar, dass Frequenzen um 50Hz herum stark in den gemessenen EEG-Daten vertreten sind. Hier hat das Headset den Wechselstrom

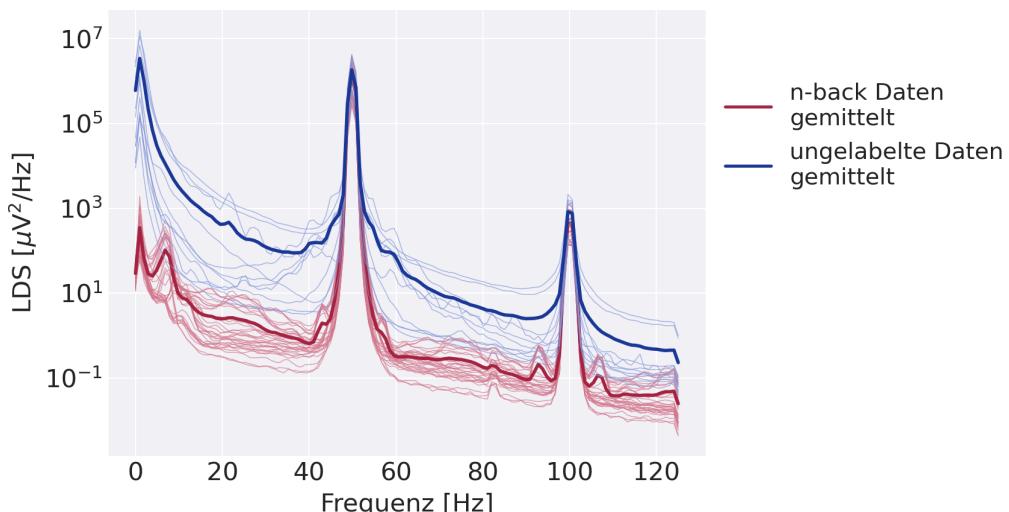


Abbildung 6: Leistungsdichtespektrum der EEG-Daten

³v1.5.0, <https://doi.org/10.5281/zenodo.7093122>

des Stromversorgungsnetzes aufgegriffen, welcher in Europa in einer Frequenz von 50Hz schwingt. Auch die deutlich vertretenen Frequenzen um 100Hz herum sind auf diese Quelle zurückzuführen. Diese Oberschwingungen werden durch elektrische Geräte der Grundschatzung des Wechselstroms in doppelter Frequenz hinzugefügt [37]. Die für die Unterscheidung der vier Stufen mentaler Auslastung nach der *n-back*-Aufgabe relevanten Frequenzbänder befinden sich im eher niedrigfrequenten Bereich. Vor allem das Alpha-Band (8-13Hz) und das Theta-Band (4-8Hz) sind wichtig, andere Frequenzen können jedoch auch noch relevante Informationen enthalten [15].

In Abbildung 6 ist außerdem erkennbar, dass das Leistungsdichtespektrum der *n-back*-Daten sich von dem der ungelabelten Daten unterscheidet. Tendenziell sind alle Frequenzen in den ungelabelten Daten stärker vertreten. Woran das liegt, ist nicht eindeutig nachvollziehbar. Grundsätzlich handelt es sich bei beiden Datensätzen weitestgehend um Aufnahmen, die im Rahmen von Computerarbeit erstellt wurden und die in dieser Hinsicht ähnlich sein sollten. Die ungelabelten Daten könnten insgesamt mehr Bewegungen des Körpers aufgegriffen haben, da die Sitzungen hier weniger kontrolliert stattgefunden haben. Auch sind die ungelabelten Aufnahmen alle deutlich länger als die gelabelten. Zeitlich sind die ungelabelten EEG-Aufnahmen eher nach denen aus der *n-back*-Aufgabe entstanden.

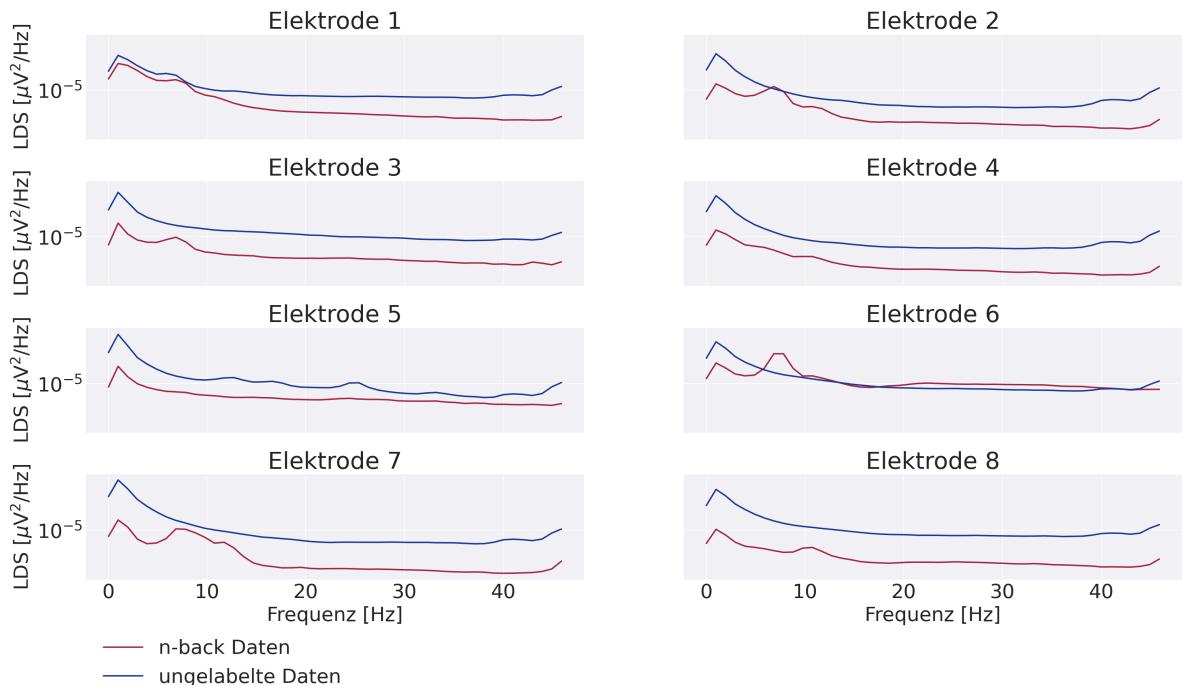


Abbildung 7: Leistungsdichtespektren der Kanäle gemittelt für die normalisierten EEG-Stücke von 2s Länge, gefiltert mit 0.1-55Hz-Bandpassfilter

In jedem Fall galt es, die ungelabelten und die *n-back*-Daten mehr einander anzugeleichen, bevor die neuronalen Netze mit den Daten lernen sollten. Auch sollten Artefakte möglichst herausgefiltert werden. Zunächst wurde dafür ein 0.1-55Hz-Bandpassfilter verwendet. Dieser entfernt in den Daten sichtbaren sehr niedrigfrequenten Drift unterhalb von 0.1Hz sowie Frequenzen oberhalb von 55Hz. Im weiteren Verlauf der Entwicklung dieser Arbeit ist aufgefallen, dass eine solche Filterung noch nicht stark genug sein könnte. Die durch das Stromnetz eingeführten Artefakte um 50Hz sind bei der Verwendung eines solchen Bandpassfilters immer noch in den Daten vertreten. Auch war dies nicht ausreichend für eine Angleichung der gelabelten und der ungelabelten EEG-Daten, wie man in Abbildung 7 erkennen kann. Die Spektren der *n-back*-Daten liegen hier immer noch unter denen der ungelabelten Daten. Für diese Abbildung wurden die Daten schon so weit aufbereitet, dass sie den neuronalen Netzen als Eingabe dienen konnten. Erst mit einem 0.1-45Hz-Bandpassfilter sehen die Leistungsdichtespektren der beiden Datensätze, wie in Abbildung 8 dargestellt, ähnlich aus.

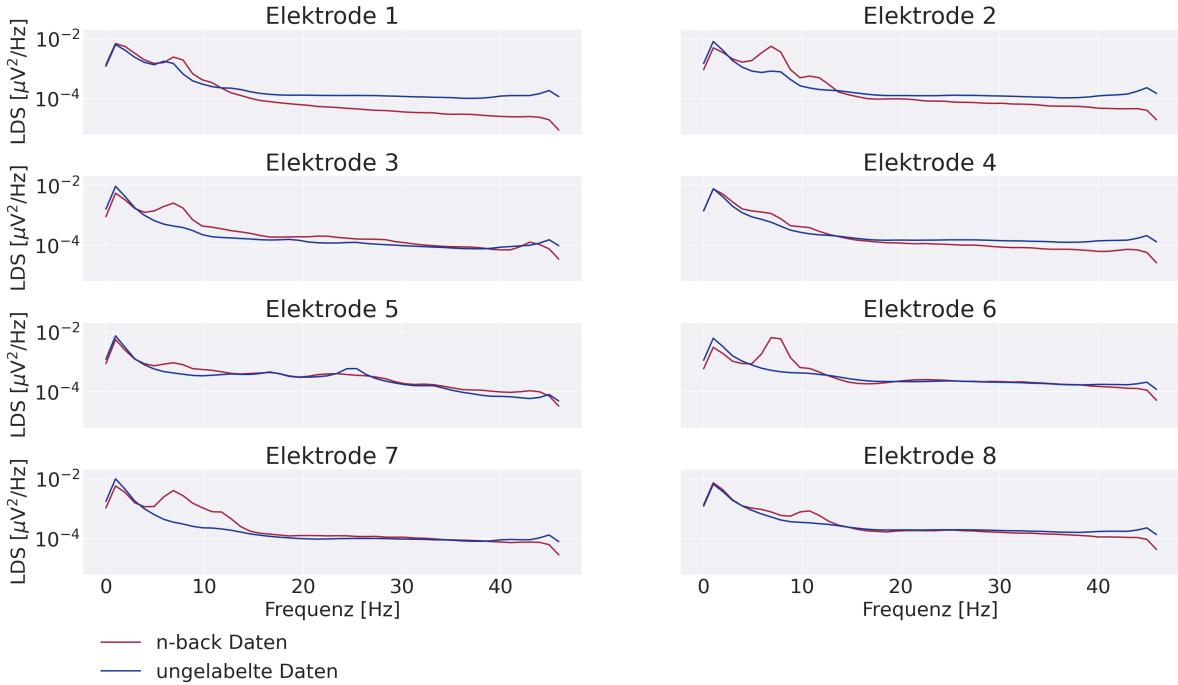


Abbildung 8: Leistungsdichtespektren der Kanäle gemittelt für die normalisierten EEG-Stücke von 2s Länge, gefiltert mit 0.1-45Hz-Bandpassfilter

Ein 0.1-45Hz-Bandpassfilter schien somit für die Aufbereitung der Daten angebrachter zu sein. Trotzdem wurden auch Modelle trainiert, bei denen die EEG-Daten mit einem 0.1-55Hz-Bandpassfilter behandelt wurden. So ließ sich ein Vergleich zwischen diesen zwei Filterungsarten anstellen, und es konnten die jeweiligen Auswirkungen, die sie auf das Training und die Ergebnisse der Modelle haben, genauer untersucht werden.

Damit die Daten den neuronalen Netzen als Eingabe dienen konnten, mussten sie in kleinere Stücke portioniert werden, die hier 2s Datenaufnahme entsprachen. Für die Erstellung dieser Samples wurde innerhalb einer EEG-Aufnahme ein 2s-Fenster in 1s-Schritten über die Aufnahme geschoben. Zwei benachbarte Samples hatten also eine Überschneidung von 1s. Diese Überlappung resultiert gegenüber einer Schrittgröße von 2s in einer größeren Menge an Samples. Es gibt Studien, die zeigen, dass größere Fenster bei der Erstellung der Samples zu besseren Klassifikationsergebnissen führen können [18], [17]. Im Sinne einer möglichen Verwendung solcher Modelle für BCI-Anwendungen sind kleinere Fenstergrößen jedoch eher wünschenswert, weswegen sich hier für 2s-Intervalle entschieden wurde. Wie beim Training von neuronalen Netzen üblich, wurden die Samples außerdem normalisiert. Dafür wurde die *Min-Max-Skalierung* verwendet, welche dafür sorgt, dass sich alle Werte eines Samples im Intervall $[0, 1]$ befinden. Eine solche Normalisierung der Eingabedaten hilft, das Training von neuronalen Netzen zu beschleunigen [38]. Es wurde auch der *RobustScaler* von *scikit-learn*⁴ [39] getestet, welcher mit Ausreißern besser umgehen können soll. Das hat aber nicht zu einer Verbesserung der Ergebnisse geführt, weswegen die simplere Min-Max-Skalierung gewählt wurde.

Bei den ungelabelten Daten wurden die ersten 30s jeder Aufnahme entfernt und nicht für die Erstellung der Samples verwendet, da zu Beginn der EEG-Aufnahmen hier oft ungewöhnliche Artefakte in den Daten erkennbar waren. Des Weiteren gibt es bei der Datenaufnahme mit dem Headset einen Kanal, der zu jedem Messzeitpunkt angibt, ob die Daten *valid* sind. Aus der Dokumentation geht nicht eindeutig hervor, was es bedeutet, wenn die Daten nicht valide sind. Vermutlich hat es in den zeitlichen Intervallen, in denen der Wert für *valid=False* ist, Verbindungsprobleme zum Headset gegeben. Dies

⁴<https://github.com/scikit-learn/scikit-learn/releases/tag/1.1.2>

ist nur selten vorgekommen, aber diese Zeitabschnitte wurden ebenso aus dem Datensatz entfernt. Darüber hinaus war es bei der Aufnahme der Daten teilweise bei den Elektroden 3 und 5 schwierig, einen konsistenten Kontakt mit der Kopfhaut herzustellen, da die Haube die Tendenz hatte, hier etwas abzustehen. Diese Kanäle waren daher besonders artefaktbehaftet, wurden aber dennoch zum Training der Modelle verwendet, da annehmbar ist, dass sie trotzdem noch relevante Informationen enthalten. So mussten die Modelle auch lernen, mit einer gewissen Menge an Artefakten in den Daten umzugehen.

5.2 EEGNet

Für die überwacht und selbstüberwacht trainierten Modelle, die Repräsentationen der EEG-Daten erlernen sollten und deswegen auch *Encoder* genannt werden, wurde die Netzwerkarchitektur des *EEGNet* [20] verwendet. Das *EEGNet* ist ein relativ kompaktes Convolutional Neural Network (siehe Abschnitt 2.3.1), das für die Anwendung in unterschiedlichen Brain Computer Interfaces entwickelt wurde. Die Implementierung des *EEGNet* ist öffentlich zugänglich. Da diese allerdings mit *tensorflow* erstellt wurde und für diese Arbeit *pytorch*⁵ [40] für die Entwicklung von neuronalen Netzen verwendet wurde, musste das *EEGNet* reimplementiert werden. Dabei wurden einige Bestandteile der Architektur angepasst. In Abbildung 9 ist eine vereinfachte Darstellung der für diese Arbeit angepassten Netzwerkarchitektur des *EEGNet* zu sehen.

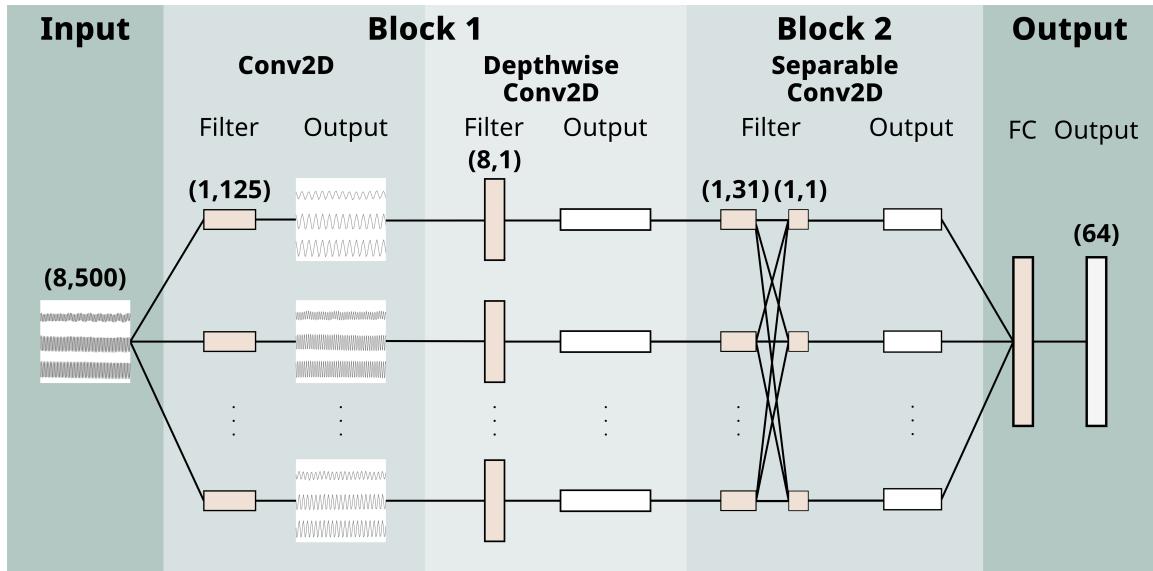


Abbildung 9: Vereinfachte Darstellung der angepassten EEGNet-Architektur

Ein EEG-Sample, das 2s Datenaufnahme entspricht, dient dem Netz als Eingabe. Bei acht Kanälen und einer Abtastrate von 250Hz hat es eine Dimensionalität von (8, 500). Im *EEGNet* durchläuft ein Sample dann in zwei Blöcken eine Reihe von Convolutions. Hierbei wurden die Filtergrößen gegenüber dem originalen *EEGNet* [20] an die hier verwendete Abtastrate und Kanalanzahl angepasst. Im ersten Block laufen zunächst Filter der Größe (1, 125) über die EEG-Kanäle, welche Frequenzinformationen ab 2Hz und höher lernen sollen. Für jeden dieser zeitlichen Filter werden bei der *Depthwise Convolution* räumliche Filter der Größe (8, 1) über die acht Kanäle hinweg erlernt. Im zweiten Block folgt dann eine *Separable Convolution*, welche aus einer *Depthwise Convolution* und einer *Pointwise Convolution* besteht. Die *Depthwise Convolution* mit einer Filtergröße von (1, 31) repräsentiert dabei etwa 500ms Informationen in den EEG-Daten. Die darauffolgende *Pointwise Convolution* soll diese Informationen möglichst optimal kombinieren. Die Ausgabe von Block 2 wird zu einem Vektor veknüpft, der zum Schluss noch durch eine vollständig verbundene Schicht läuft und so in einem Ausgabevektor

⁵<https://github.com/pytorch/pytorch/releases/tag/v1.12.1>

der Größe 64 resultiert. Dieser stellt dann eine Repräsentation des ursprünglichen EEG-Samples dar, welches als Eingabe gedient hat. Neben den Convolutions enthält das EEGNet natürlich noch Pooling-Layer und weitere Schichten. Für einen detaillierten Aufbau des Netzes siehe Abschnitt A.

5.3 Siamesisches Netzwerk

Mit Siamesischen Netzwerken lässt sich ein kontrastiver Ansatz des maschinellen Lernens verfolgen, bei dem Merkmale der Daten aufgrund von Ähnlichkeiten und Unterschieden zwischen den Samples gelernt werden. Hierbei dienen mehrere Samples jeweils als Eingabe für identische Netze, die alle die gleichen Gewichte haben, und diese Samples werden miteinander verglichen. In der Implementierung solcher Netzwerke handelt es sich dabei tatsächlich in der Regel nur um ein Netzwerk, das direkt nacheinander die einzelnen zu kontrastierenden Samples verarbeitet. Die von einem Netzwerk ausgegebene Repräsentation eines Samples wird durch die verwendete Loss-Funktion mit den Repräsentationen der anderen Samples verglichen. Das Netz soll dabei lernen, ähnliche Repräsentationen für ähnliche EEG-Samples und möglichst unterschiedliche Repräsentationen für sich stark voneinander unterscheidende EEG-Samples zu generieren.

Diese Methodik des kontrastiven maschinellen Lernens wurde sowohl für die überwacht als auch für die selbstüberwacht lernenden Encoder genutzt. Die verwendete Netzwerkarchitektur war die des bereits beschriebenen EEGNet [20].

5.3.1 Triplet Loss

Triplet Loss war die Loss-Funktion, die beim Training der Encoder genutzt wurde und die es im Laufe des Trainings der Modelle zu minimieren galt. Ein Triplet besteht bei dieser kontrastiven Loss-Funktion aus drei Samples, dem *Anker*, dem *positiven Sample* und dem *negativen Sample*. Beim überwachten Lernen, wobei die einzelnen Samples Klassen angehören, stammen der Anker und das positive Sample aus derselben Klasse, sie bilden das positive Paar. Das negative Sample stammt aus einer anderen Klasse, wobei Anker und negatives Sample das negative Paar bilden. In diesem Fall unter Verwendung des n -back-Datensatzes hieß das zum Beispiel, dass das positive Paar der Klasse $n = 1$ angehörte und das negative Sample aus der Klasse $n = 3$ stammte. Wie in Abbildung 10 dargestellt, durchlaufen alle drei Samples nun das neuronale Netz, was in drei Repräsentationen der Samples resultiert. Triplet-Loss vergleicht nun die Distanzen zwischen den Repräsentationen im Merkmalsraum, der durch die Ausgabe des Netzes gebildet wird. Dabei soll die Distanz zwischen Anker und positivem Sample möglichst minimiert werden, sie sollen nahe beieinander angeordnet werden. Gleichzeitig soll die Distanz zwischen Anker und negativem Sample maximiert werden, damit sie im Merkmalsraum möglichst weit voneinander entfernt sind. Insgesamt soll dies beim überwachten Lernen dazu führen, dass Samples

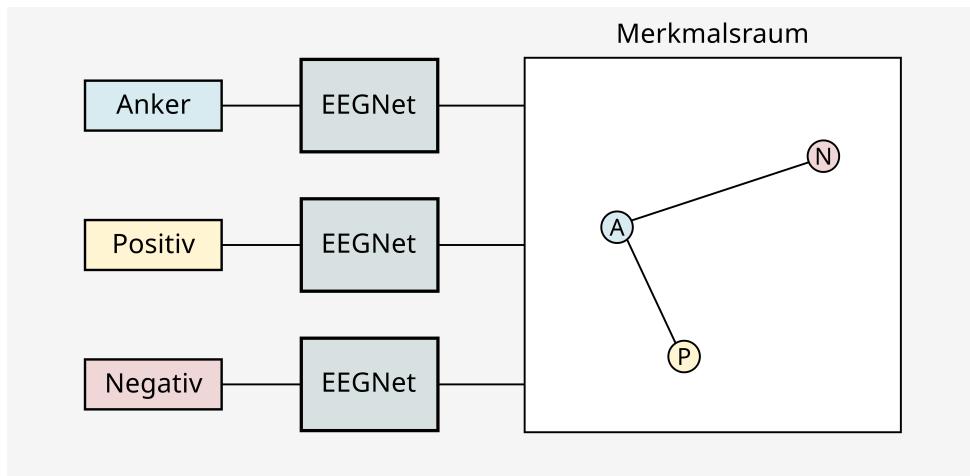


Abbildung 10: Triplet-Loss Visualisierung

aus einer Klasse im Merkmalsraum nahe beieinander und möglichst getrennt von den Samples anderer Klassen abgebildet werden. In dem Beispiel in Abbildung 10 ist der Merkmalsraum, um ihn grafisch darstellen zu können, nur zwei-dimensional. Die Ausgabe der Encoder ist 64-dimensional, die Distanzberechnung im Rahmen von Triplet-Loss funktioniert in so einem höherdimensionalen Raum jedoch gleich.

Beim selbstüberwachten Lernen wurde auf ungelabelten EEG-Daten trainiert, hier ließen sich die Samples somit keinen Klassen zuordnen. Den Anker bildete hier ein EEG-Sample aus dem ungelabelten Datensatz, und das negative Sample war ein davon unterschiedliches zufällig ausgewähltes Sample. Das positive Sample war eine augmentierte, leicht veränderte Version des Ankers. Das heißt, dass das positive Sample grundsätzlich dem Anker ähnlich war, sich durch die hinzugefügten Augmentationen aber von ihm unterschieden hat. Auch bei den selbstüberwachten Encodern galt, dass die Modelle lernen sollten, Anker und positives Sample nahe beieinander im Merkmalsraum und das negative Sample vom Anker entfernt abzubilden. Die Modelle mussten erkennen, dass ein augmentiertes Sample trotz der hinzugefügten Augmentationen ähnliche Eigenschaften aufwies wie das originale. Das Ziel hierbei war, dass die selbstüberwacht trainierten Modelle lernen, sinnvolle Repräsentationen der EEG-Daten zu generieren, die die wichtigen Merkmale der Daten gut zusammenfassen. Diese Repräsentationen sollten sich dann im Sinne der Anwendungsaufgabe weiterverwerten lassen. In diesem Fall wurden die erlernten Repräsentationen im Anschluss evaluiert, indem auf ihnen ein Klassifikationsnetz trainiert wurde, das EEG-Daten aus dem n -back-Datensatz ihren Klassen zuordnen sollte.

Es gibt mehrere Optionen, zwei Vektoren im Merkmalsraum miteinander zu vergleichen, um dieses Maß als Distanzfunktion bei Triplet-Loss zu verwenden. Hier wurde die *euklidische Distanz* zwischen den beiden Repräsentations-Vektoren verwendet. Ebenso war zu Beginn des Projektes die *Kosinus-Ähnlichkeit* getestet worden. Diese schien jedoch in den ersten durchgeführten Tests zu schlechteren Ergebnissen im Vergleich zu der euklidischen Distanz zu führen und wurde daher wieder verworfen. Des Weiteren wird im Rahmen von Triplet-Loss noch ein Parameter für eine Marge definiert, die zwischen positivem und negativem Paar hergestellt werden soll. Das negative Sample soll dadurch nicht nur grundsätzlich weiter vom Anker entfernt sein als das positive Sample, durch die Marge soll die Entfernung zum negativen Sample noch weiter erhöht werden.

Triplet-Loss mit der Repräsentation des Ankers A , des positiven Samples P , des negativen Samples N und mit einer Marge $\alpha > 0$ ist damit wie folgt definiert:

$$L(A, P, N) = \max\{\|A - P\|_2 - \|A - N\|_2 + \alpha, 0\}$$

5.4 Klassifikation

Die Ausgabevektoren der überwacht und selbstüberwacht trainierten Encoder für sich lassen zunächst noch keine Aussage über die Klassenzugehörigkeit eines Samples zu. Um ein eingegebenes EEG-Sample klassifizieren zu können, bedurfte es noch weiterer Schritte. Einerseits fand eine Klassifikation im Merkmalsraum mit k -nearest-neighbours statt (siehe Abschnitt 5.4.1). Zusätzlich dazu wurden im Anschluss an die Entwicklung der Encoder noch Klassifikationsnetze trainiert (siehe Abschnitt 5.4.2).

5.4.1 K-nearest-neighbours

K -nearest-neighbours (knn) wurde verwendet, um im Merkmalsraum, der durch die Ausgabe der Encoder jeweils gebildet wird, eine Klassifikation der EEG-Samples durchzuführen. Es handelt sich hier um eine Methode des überwachten Lernens. Bei diesem Algorithmus benötigt man Samples aus dem n -back-Datensatz, deren Klassenzugehörigkeit bekannt ist. Diese durchlaufen einen Encoder, um dann die Lage der Repräsentationen im Merkmalsraum zu betrachten. Möchte man ein EEG-Sample, dessen Klassenzugehörigkeit unbekannt ist, in eine n -back-Klasse einordnen, schaut man sich die Anordnung dieses Samples im Merkmalsraum des Encoders im Vergleich zu den Samples mit bekannten Klassen

an. Man betrachtet die k nächsten Nachbarn des unzugeordneten Samples und schaut, welche Klasse bei diesen Nachbarn am häufigsten vorkommt. Dieser Klasse wird dann auch das zu klassifizierende Sample zugeordnet. Dafür muss für jedes zu klassifizierende Sample die Distanz zu allen Samples mit bekannter Klasse berechnet werden. Die k Samples, zu denen die Distanz am geringsten ist, sind dann die gesuchten Nachbarn. Hier wurde wie bei Triplet-Loss die euklidische Distanz zur Abstandsberechnung verwendet.

In dem Beispiel in Abbildung 11 ist $k = 5$, es werden folglich die fünf nächsten Nachbarn des zu klassifizierenden Samples betrachtet. Davon stammen drei Samples aus der Klasse $n = 1$ und zwei aus der Klasse $n = 0$. Folglich wird das zu klassifizierende Sample der Klasse $n = 1$ zugeordnet.

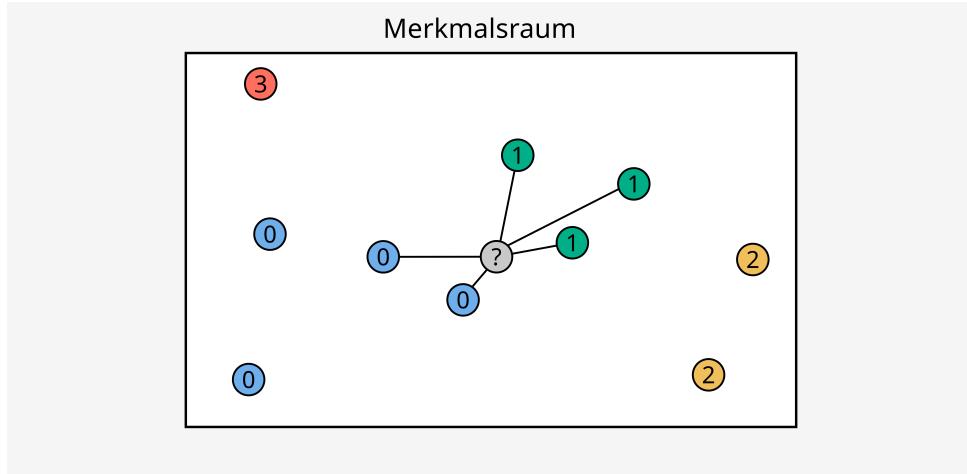


Abbildung 11: Beispiel für 5-nearest-neighbours

Diese Art des Klassifizierens ist sinnvoll in der Anwendung bei den überwacht trainierten Modellen. Triplet-Loss sollte bei diesen Modellen dafür sorgen, dass sich Samples aus derselben Klasse nahe beieinander und entfernt von den Samples anderer Klassen im Merkmalsraum befinden. KnN sollte hier somit gut funktionieren. Bei den selbstüberwachten trainierten Modellen hingegen konnte man nicht davon ausgehen, dass eine Klassifikation mit knn zu guten Ergebnissen führen würde. Die selbstüberwachte Lernaufgabe, bei der die Triplets mit Augmentationen gebildet wurden und Klassenzugehörigkeit keine Rolle spielte, bereitete ein Modell nicht darauf vor, Samples in Stufen unterschiedlicher mentaler Auslastung im Rahmen der n -back-Aufgabe zu klassifizieren. Zur Evaluation der selbstüberwachten trainierten Encoder und der Verwendbarkeit der erlernten Repräsentationen für eine Klassifikation von n -back-Daten brauchte es somit noch einen anderen Weg des Klassifizierens.

5.4.2 Klassifikationsnetz

Um die Verwendbarkeit der erlernten Repräsentationen zu überprüfen, wurden für die Encoder im Anschluss jeweils Klassifikationsnetze trainiert. Diese bekamen die 64-dimensionale Ausgabe eines Encoders als Eingabe und sollten direkt die jeweilige n -back-Klasse des Samples ausgeben. Trainiert wurden diese Netze mit dem n -back-Datensatz. Die Architektur des Klassifikationsnetzes war ein einfacher gehaltenes MLP. Es handelte sich hier um lediglich drei vollständig verbundene Schichten mit 128, 64 und vier Neuronen. Zwischen den Schichten gab es eine *ReLU*-Aktivierung, die Ausgabe der letzten Schicht durchlief eine *Softmax*-Aktivierung. Die Ausgabe der vier Klassen erfolgte dabei in der *One-Hot-Kodierung*. Der Ausgabevektor lässt sich so interpretieren, dass jeder der vier Werte eine Wahrscheinlichkeit dafür darstellt, dass das Sample zu einer der Klassen gehört. Das Sample wurde dann der Klasse zugeordnet, bei der die vorhergesagte Wahrscheinlichkeit am höchsten war. Die verwendete Loss-Funktion beim Training der Netze war *Kreuzentropie*.

Diese Klassifizierungsmethodik war vor allem notwendig, um zu überprüfen, ob sich die selbstüberwacht erlernten EEG-Repräsentationen für eine Klassifikation unterschiedlicher Stufen mentaler Auslastung sinnvoll weiterverwenden lassen. Um die selbstüberwacht lernenden Encoder besser mit den überwacht lernenden vergleichen zu können, wurden auch für die überwacht trainierten Modelle Klassifikationsnetze entwickelt.

5.5 Training

In diesem Kapitel soll es darum gehen, wie das überwachte und selbstüberwachte Training der Encoder (siehe Abschnitt 5.5.1 und Abschnitt 5.5.2) sowie das Training der Klassifikationsnetze (siehe Abschnitt 5.5.3) bezüglich der Lernmethodik, der jeweils verwendeten Trainingsdaten und der gewählten Hyperparameter durchgeführt wurde. Implementiert wurden die neuronalen Netze und das Training derselben mit *pytorch* [40].

5.5.1 Überwacht

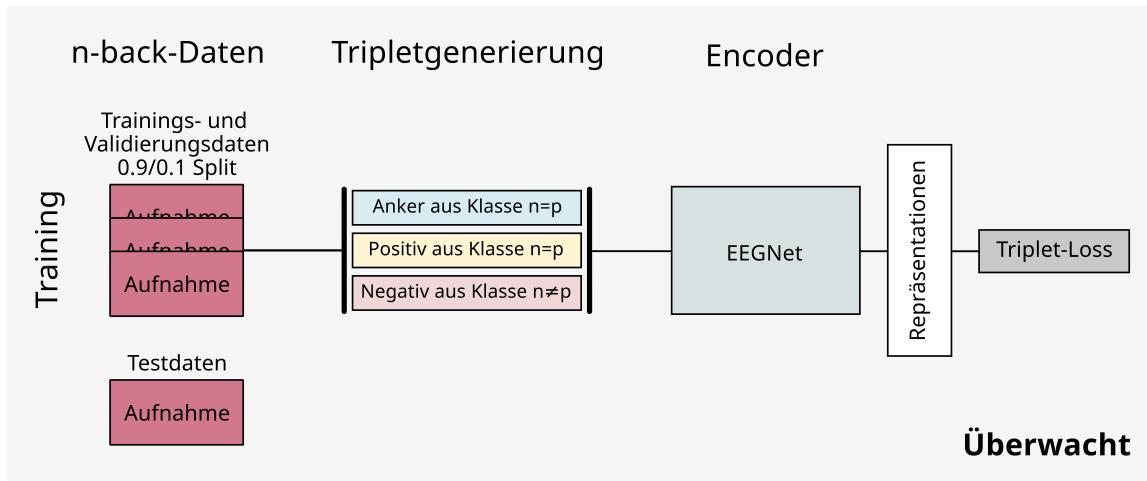


Abbildung 12: Training der überwacht lernenden Encoder

Die überwacht lernenden Encoder wurden auf den gelabelten EEG-Daten trainiert, die mit der n -back-Aufgabe erstellt worden waren. Die Daten lassen sich vier Klassen zuordnen: $n \in \{0, 1, 2, 3\}$. Diese Klassen stehen für unterschiedliche Stufen mentaler Auslastung, je höher der Wert von n , desto höher die Auslastung. Die auf dem EEGNet [20] basierenden Modelle lernten kontrastiv unter der Verwendung von Triplet-Loss als zu minimisierende Loss-Funktion im Merkmalsraum Samples gleicher Klasse nahe beieinander anzugeordnen.

Der n -back-Datensatz beinhaltet vier EEG-Aufnahmen, wovon immer drei für Training und Validierung und eine zum Testen der Modelle genutzt wurden. Dabei wurden mehrere Modelle trainiert, so dass jede der vier EEG-Aufnahmen in einer Konfiguration mal Testaufnahme war. Dies resultierte in vier Konfigurationen für das Aufteilen in Trainings- und Testaufnahmen, wobei dann jeweils zwei Modelle pro Konfiguration trainiert wurden. Ein Teil der Samples aus den drei zum Training verwendeten Aufnahmen wurde zur Validierung verwendet, der Anteil betrug 10% und wurde zufällig ausgewählt, wobei darauf geachtet wurde, dass alle Klassen und Aufnahmen in den Validierungsdaten zu gleichen Anteilen vertreten waren. Bei zwei Modellen mit gleicher Konfiguration bezüglich der Trainings- und Testaufnahmen wurden die für die Validierung verwendeten Daten jeweils neu bestimmt. Die EEG-Daten wurden im Vorhinein entweder mit einem 0.1-45Hz-Bandpassfilter oder mit einem 0.1-55Hz-Bandpassfilter aufbereitet. Insgesamt wurden somit acht Encoder pro Filterungsart überwacht trainiert.

Die Validierungsdaten wurden nicht zum Training der Modelle verwendet, dienten aber dazu, dieses zu überwachen. Zusätzlich wurden die Validierungsergebnisse beim Anpassen der Hyperparameter

genutzt. Für die anschließende Evaluation wurden die Modelle verwendet, bei denen der Validierungs-Loss der Trainingsepochen im Vergleich zu den restlichen Epochen am niedrigsten war. Durch diese Methode des *Early Stoppings* soll eine Überanpassung auf die Trainingsdaten vermieden werden.

Vor jeder Trainingsepoke wurden die Triplets neu generiert. Jedes Sample aus dem Trainingsdatensatz war ein mal Anker, und es wurden für jeden dieser Anker alle Samples aus der gleichen Klasse genutzt, um alle möglichen positiven Paare zu generieren. Für jedes positive Paar wurde dann zufällig ein negatives Sample gewählt, das aus einer anderen Klasse stammte. Die Validierungs-Triplets wurden auf die gleiche Weise erstellt.

Es fand ein Tuning der Hyperparameter statt, wobei sich an der von Godbole et al. [41] beschriebenen Methodik orientiert wurde. Die Klassifizierungs-Accuracy mit knn auf den Validierungsdaten wurde dafür genutzt, um unterschiedliche getestete Hyperparameter zu evaluieren und die zu finden, die die besten Ergebnisse erzielen. Getestet wurden dabei verschiedene Werte für die Lernrate, die Ausgabedimension der Encoder sowie der im Rahmen von Triplet-Loss gewählten Marge. Das Tuning wurde dabei sicherlich nicht in vollem Maße ausgereizt, da dies zu viel Zeit in Anspruch genommen hätte. Die besten Ergebnisse ließen sich mit einer Lernrate von 0.001, einer Ausgabedimension von 64 und einer Marge von 250 erzielen.

5.5.2 Selbstüberwacht

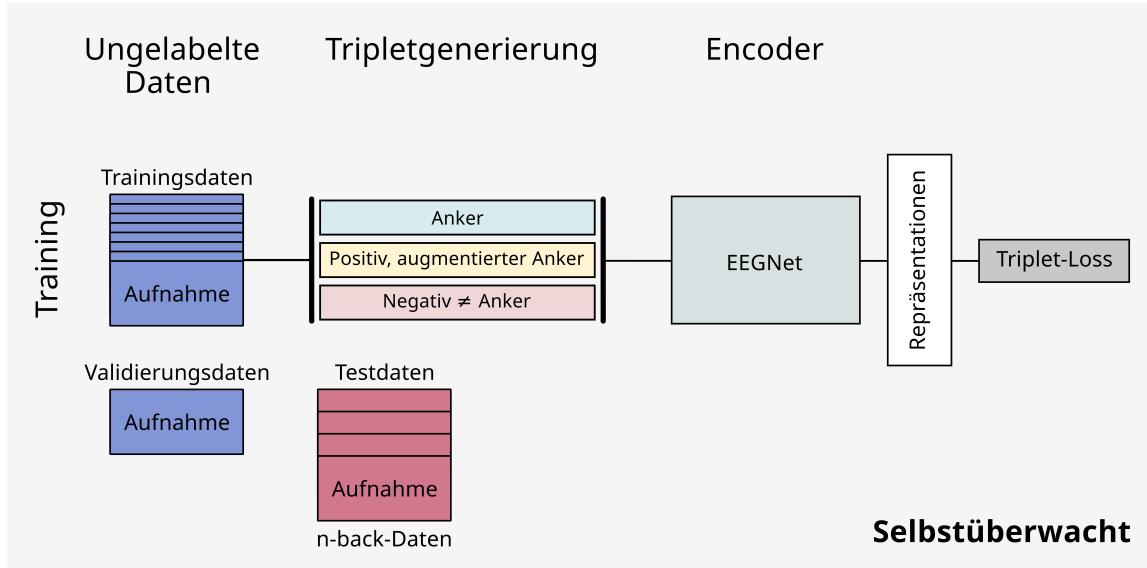


Abbildung 13: Training der selbstüberwachten lernenden Encoder

Die selbstüberwachten lernenden Encoder wurden auf den ungelabelten EEG-Daten trainiert. Wie die überwachten trainierten Encoder haben sie kontrastiv mit Triplet-Loss als die zu minimierende Loss-Funktion gelernt. Die Triplets wurden mithilfe von Augmentationen erstellt (siehe Abschnitt 5.5.2.1). Das Ziel beim selbstüberwachten Lernen war, dass die Modelle sinnvolle Repräsentationen der Daten erlernen, die relevante Merkmale von EEG-Daten beinhalten. Diese Repräsentationen sollten sich dann für nachgelagerte Aufgaben weiterverwenden lassen. Hier wurden sie im Anschluss beim Training der Klassifikationsnetze für die Klassifikation von *n*-back-Daten genutzt.

Der ungelabelte Datensatz umfasst neun EEG-Aufnahmen, wovon immer acht zum Training und eine für die Validierung genutzt wurden. Dabei gab es zwei ausgewählte Aufnahmen, die potenziell als Validierungsaufnahme dienen konnten und die jeweils etwa 10% der Gesamtmenge an ungelabelten Daten darstellten. Die Testdaten bildeten hier die gelabelten *n*-back-Daten. Vor jeder Trainingsepoke wurden die Triplets neu erstellt, wobei die positiven Samples der Daten-Triplets hier augmentierte Versionen des Ankers darstellten. Dabei wurden drei Stufen bezüglich der Stärke der Augmentationen

getestet. Pro Augmentationsfaktor wurden zwei Modelle trainiert, wobei jeweils eine andere EEG-Aufnahme für die Validierung genutzt wurde. Darüber hinaus liefern auch hier die EEG-Daten vor dem Training entweder durch einen 0.1-45Hz-Bandpassfilter oder durch einen 0.1-55Hz-Bandpassfilter. Insgesamt sind folglich sechs Encoder pro Filterungsart selbstüberwacht trainiert worden.

Die negativen Samples wurden zufällig ausgewählt, wobei sie aus derselben EEG-Aufnahme stammten wie der Anker. Dies sollte verhindern, dass die Modelle lediglich lernen Samples aus anderen Aufnahmen voneinander abzugrenzen. Ohne die Bedingung der gleichen Aufnahmезugörigkeit wäre die Wahrscheinlichkeit, dass Anker und negatives Sample aus unterschiedlichen Aufnahmen stammen, relativ hoch. Darüber hinaus wurde sichergestellt, dass beim negativen Paar eine zeitliche Differenz zwischen den Samples von mindestens 6s besteht, damit sich Anker und negatives Sample nicht zu ähnlich sind.

Die Hyperparameter waren zunächst dieselben wie die, die beim überwachten Lernen gewählt wurden. Hier wurde kein ausführliches Tuning durchgeführt, aber es fanden einige Tests mit variierenden Werten für die Hyperparameter statt. Letztendlich schienen aber eine Ausgabedimension von 64 und eine Marge von 250 bei Triplet-Loss auch hier auf den Validierungsdaten gut zu funktionieren, nur die Lernrate wurde angepasst. Dabei wurde die Beobachtung gemacht, dass bei höherem Augmentationsfaktor eine kleinere Lernrate besser funktioniert hat. Bei einem Augmentationsfaktor von 1 betrug die Lernrate 0.1, bei einem Faktor von 2 hatte die Lernrate einen Wert von 0.05, und bei dem höchsten Augmentationsfaktor 3 betrug die Lernrate 0.01.

Der hier verwendete Ansatz des kontrastiven selbstüberwachten maschinellen Lernens mit augmentierten EEG-Daten ist dem von Mohsenvand et al. [14] nachempfunden. In der konkreten Methodik gibt es jedoch einige Unterschiede zu dieser Arbeit. Unter anderem wurden unterschiedliche Netzarchitekturen genutzt, und die verwendeten EEG-Daten stammten aus anderen Domänen. Auch wurden die Encoder in der Vorbildstudie für einen EEG-Kanal entwickelt, während in dieser Arbeit alle Kanäle zusammen betrachtet wurden.

5.5.2.1 Augmentationen

Die positiven Samples eines Triplets wurden durch augmentierte Versionen des Ankers dargestellt. Insgesamt wurden fünf verschiedene Augmentationen angewendet, die in Abbildung 13 dargestellt sind. Bei der Erstellung eines positiven Samples wurden davon zufällig zwei Augmentationen ausgewählt, die auf das EEG-Signal des Ankers angewendet wurden. Die vier Augmentationen Zeitverschiebung, Maskieren, Bandstopfilterung und das Hinzufügen von Rauschen stammten aus der bereits genannten Vorbildstudie [14]. Diese kam auch zu dem Schluss, dass eine Kombination aus mehreren Augmentationen beim selbstüberwachten Lernen besser funktioniert als die Verwendung von nur einzelnen Augmentationen. Dort wurden zusätzlich noch zwei weitere Augmentationen verwendet, hierbei handelte es sich um eine Spannungsverschiebung entlang der y-Achse sowie um eine Skalierung der Amplitude. Diese Augmentationen würden hier bei der angewandten Normalisierung der Daten jedoch ihren Effekt verlieren.

Bei der Zeitverschiebung wird das Signal entlang der x-Achse nach rechts oder links verschoben. Ein maskiertes Signal wird anschließend an die Min-Max-Normalisierung in einem zufällig ausgewählten zeitlichen Intervall auf 0.5 gesetzt. Bei der Augmentation mit einem Bandstopfilter wird ein solcher Filter mit einer Breite von 5Hz für ein zufälliges Frequenzintervall auf dem Signal angewendet. Eine weitere Augmentation ergibt sich, indem dem Signal gaußsches Rauschen hinzuzufügt wird. Die fünfte Augmentation bildet die Frequenzverschiebung, vorgeschlagen von Rommel et al. [42]. Hier werden die Daten unter Verwendung der Bibliothek *scipy*⁶ [43] zunächst mit einer schnellen Fourier-Transformation in ihre Frequenzanteile zerlegt. Dann wird eine Verschiebung der Frequenzen des Signals

⁶<https://github.com/scipy/scipy/releases/tag/v1.9.1>

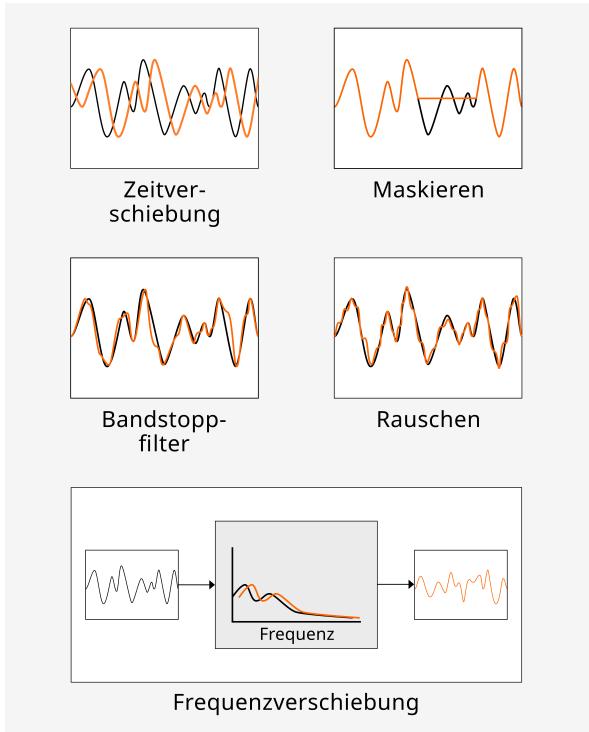


Abbildung 14: Augmentationen (orange) eines EEG-Signals (schwarz)

durchgeführt. Mit der inversen schnellen Fourier-Transformation findet anschließend eine Umwandlung zurück zu einem zeitdiskreten Signal statt. Diese Augmentation wurde zusätzlich zu den anderen vier eingeführt, um verstärkt zu fördern, dass die selbstüberwacht erlernten Repräsentationen Frequenzmerkmale der EEG-Daten enthalten, die vermutlich besonders relevant sind.

Es wurden drei Augmentationsfaktoren getestet. Die am wenigsten starken Augmentationen mit einem Augmentationsfaktor von 1 entsprechen den von Mohsenvand et al. [14] verwendeten Stärken, die relevanten Wertebereiche dafür sind in Tabelle 1 angegeben. Bei den Augmentationsfaktoren 2 und 3 werden die Maximalwerte der jeweiligen Wertebereiche mit den Faktoren multipliziert, um stärkere Augmentationen zu erhalten. Nur bei dem Bandstoppfilter wird stattdessen die Breite des Filters mit dem Augmentationsfaktor multipliziert.

Augmentation	Min	Max	zusätzliche Bedingungen
Zeitverschiebung [s]	0.04	0.24	in beide Richtungen
Maskieren [s]	0.08	0.72	
Bandstoppfilter [Hz]	3	83	Breite 5Hz
Gaußsches Rauschen [σ]	0.02	0.2	
Frequenzverschiebung [Hz]	-	2	

Tabelle 1: Wertebereiche der Parameter für die Augmentationen bei Augmentationsfaktor 1

5.5.3 Klassifikationsnetz

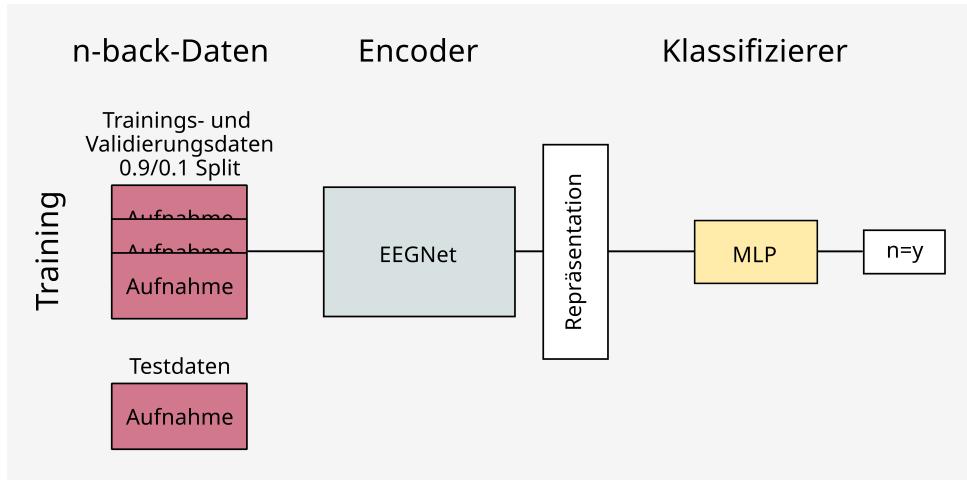


Abbildung 15: Training der Klassifikationsnetze

Die Klassifikationsnetze wurden jeweils für einen vorherig entwickelten überwacht oder selbstüberwacht lernenden Encoder auf dem *n*-back-Datensatz trainiert. Diese Klassifizierer bekamen eine von

einem Encoder stammende Repräsentation eines EEG-Samples als Eingabe und sollten das Label des Samples vorhersagen. Sie dienten vor allem der Evaluation der selbstüberwacht trainierten Encoder. Mit den Klassifikationsnetzen sollten die erlernten Repräsentationen hinsichtlich ihrer Verwendung für eine Klassifikation unterschiedlicher Stufen mentaler Auslastung überprüft werden. Um eine bessere Vergleichbarkeit mit den überwacht lernenden Encodern zu gewährleisten, wurden auch für diese Modelle Klassifikationsnetze trainiert.

Die Aufteilung in Trainings-, Validierungs- und Testdaten fand bei den Klassifikationsnetzen auf die gleiche Weise wie bei den überwacht trainierten Encodern statt. Drei gelabelte EEG-Aufnahmen wurden für Training und Validierung verwendet, eine separate Aufnahme bildete die Testdaten.

Aus zeitlichen Gründen wurden nicht für alle Encoder Klassifizierer entwickelt, sondern nur für ausgewählte Modelle. Da insgesamt keine großen Schwankungen zwischen den Ergebnissen von Modellen beobachtet wurden, die auf die gleiche Weise entwickelt wurden und lediglich unterschiedliche Datenkonfigurationen hatten, sollten die Resultate dieser Teilmenge an Modellen jedoch trotzdem über eine ausreichende Aussagekraft verfügen. Es wurden insgesamt vier überwacht trainierte Encoder ausgewählt, bei zweien wurden die EEG-Daten mit einem 0.1-45Hz-Bandpassfilter, bei den anderen beiden wurden sie mit einem 0.1-55Hz-Bandpassfilter aufbereitet. Bei den selbstüberwacht lernenden Encodern wurden für sechs Modelle Klassifizierer trainiert. Auch hier wurden die Daten bei jeweils der Hälfte der Modelle mit dem stärkeren oder schwächeren Bandpassfilter bearbeitet. Pro Filterungsart der Daten wurde für jeden der drei Augmentationsfaktoren ein selbstüberwachter Encoder ausgewählt. Für jeden selbstüberwachten Encoder wurden zwei Klassifikationsnetze mit unterschiedlichen Trainings- und Testaufnahmen trainiert. Für die überwachten Encoder wurde nur jeweils ein Klassifizierer entwickelt, hier war die Aufteilung in Trainings- und Testaufnahmen die gleiche wie schon beim Training des jeweiligen Encoders. Die beiden zum Testen verwendeten gelabelten EEG-Aufnahmen bei den Klassifikationsnetzen sind die Aufnahmen Nr. 1 und Nr. 4.

6 Evaluation

6.1 Metriken

Zur Evaluation der überwacht und selbstüberwacht trainierten Encoder wird eine Reihe an Metriken verwendet. Für die Auswertung werden dabei immer die Modelle aus der Epoche verwendet, in welcher der Validierungs-Loss während des Trainings am niedrigsten gewesen ist.

Zum einen soll die Abbildung der Daten-Triples im Merkmalsraum, der durch die Ausgabe der Modelle gebildet wird, untersucht werden. Triplet Loss (siehe Abschnitt 5.3.1) sollte beim Training dazu führen, dass der Anker und das positive Sample im Merkmalsraum nahe beieinander sind, während das negative Sample weiter vom Anker entfernt sein sollte. Die *Triplet-Accuracy* beschreibt nun den Anteil der Fälle, bei denen die Distanz des positiven Paars geringer ist als die des negativen, das positive Sample dem Anker also näher ist als das negative Sample. Beim Training wurde hier zusätzlich noch eine zu erreichende Marge im Rahmen der Loss-Funktion definiert. Das angestrebte Ziel war nicht nur, dass die Distanz des negativen größer ist als die des positiven Paars, sondern dass sie um eine Marge von 250 größer ist. In der *Triplet-Accuracy* wird diese Marge weggelassen, das macht die Ergebnisse einfacher interpretierbar. So lässt sich sagen, dass bei einer zufälligen Verteilung die *Triplet-Accuracy* 50% betragen sollte. Die Ergebnisse der trainierten Modelle sollten also mindestens besser als dieser Wert sein.

Dabei ist zu berücksichtigen, dass sich das gestellte Problem im überwachten und selbstüberwachten Fall unterscheidet, auch wenn in beiden Fällen mit Triplet Loss gearbeitet wird. Im überwachten Fall (siehe Abschnitt 5.5.1) wird auf den gelabelten n -back-Daten trainiert, wobei eine separate n -back-Aufnahme zum Testen verwendet wird. Die Validierungsdaten stammen aus denselben EEG-Aufnahmen

wie die Trainingsdaten, sind aber natürlich unterschiedlich von ihnen. Das positive Paar kommt aus derselben Klasse, das negative Sample aus einer anderen. Bei den selbstüberwacht trainierten Encodern (siehe Abschnitt 5.5.2) hingegen wird auf der größeren Menge an ungelabelten Daten trainiert und validiert, wobei die Validierungsdaten aus einer separaten Aufnahme bestehen. Die Testdaten sind hier die n -back-Daten. Die Triplets werden mithilfe von Augmentationen gebildet, wobei das positive Sample eine augmentierte Version des Ankers ist. Das negative Sample ist vom Anker unterschiedlich und wird zufällig ausgewählt.

Aufgrund der unterschiedlichen Lernaufgaben und zur besseren Unterscheidung sollen die Metriken, die die *Triplet-Accuracy* (*TA*) im überwachten und selbstüberwachten Fall beschreiben, verschiedene Namen haben. Im überwachten Fall, wo mit gelabelten Daten gearbeitet wird, soll die entsprechende Metrik *n-back-TA* heißen. Im selbstüberwachten Fall werden die Triplets mit augmentierten Daten erstellt, daher soll die Metrik hier *aug-TA* genannt werden.

Die Triplet-Accuracy eignet sich gut, um zu beurteilen, ob die Encoder gelernt haben, die konkret an sie gestellte Aufgabe zu bewältigen. Wie gut die eigentliche Klassifikation der Daten bezüglich der durch die n -back-Aufgabe erstellten Stufen an mentaler Auslastung funktioniert, soll natürlich auch untersucht werden. Dafür wird hier die normale Accuracy berechnet, die angibt, welcher Anteil der Samples richtig klassifiziert wird. Hier lässt sich einerseits die Klassifikation im Merkmalsraum mit k -nearest-neighbours betrachten (siehe Abschnitt 5.4.1), wobei hier $k = 7$ ist. Die entsprechende Metrik soll *knn-Accuracy* (*knn-A*) heißen. Des Weiteren lassen sich die Ergebnisse der Klassifikationsnetze (siehe Abschnitt 5.5.3) untersuchen, die durch die *head-Accuracy* (*head-A*) beschrieben werden sollen. Ein Klassifikationsnetz bekommt dabei als Eingabe eine Repräsentation der EEG-Daten, die von einem überwacht oder selbstüberwacht trainierten Encoder ausgegeben wird. Ein solcher Klassifizierer wurde dabei spezifisch für jeweils einen Encoder trainiert und gibt direkt eine vorhergesagte Klasse für ein Sample zurück. Bei vier möglichen Klassen ergibt sich bei einer zufälligen Zuordnung der Samples zu den Klassen eine Accuracy von 25%. Bezuglich der *knn-Accuracy* und der *head-Accuracy* sollten die Ergebnisse der Modelle diesen Wert somit übertreffen können.

Neben den Möglichkeiten der Berechnung der Accuracy, die Aussagen über den Anteil an korrekt klassifizierten EEG-Samples zulassen, lässt sich die Metrik der *mittleren absoluten Abweichung*, im Englischen *Mean Absolute Error* (*MAE*), nutzen, um auch beurteilen zu können, wie weit die Klassifikationen im Durchschnitt daneben liegen. Bei einer zufälligen Verteilung sollte dieser Wert bei vier Klassen bei 1.25 liegen. Auch der MAE wurde hier für die Klassifikation mit knn und Klassifikationsnetz bestimmt und heißt entsprechend *knn-MAE* und *head-MAE*.

Bei den selbstüberwacht trainierten Encodern wird knn in der Evaluation bei jedem Modell mehrfach angewendet. Dabei ist jede der insgesamt vier EEG-Aufnahmen aus der n -back-Anwendung ein mal der zu klassifizierende Datensatz, und die verbleibenden drei sind die Datensätze, welche die Nachbarn mit bekannter Klasse enthalten. Die vier resultierenden Ergebnisse werden dann gemittelt.

6.2 Ergebnisse

Es folgt nun die Auswertung der Ergebnisse für die überwacht (Abschnitt 6.2.1) und selbstüberwacht (Abschnitt 6.2.2) trainierten Encoder sowie die im Anschluss entwickelten Klassifikationsnetze (Abschnitt 6.2.3). In der Annahme, dass es mit diesem Filter besser gelingt, Artefakte in den Daten zu entfernen und die EEG-Aufnahmen aneinander anzulegen, liegt der Fokus der Auswertung auf den Modellen, bei denen die EEG-Daten vor dem Training mit einem 0.1-45Hz-Bandpassfilter aufbereitet wurden. Zusätzlich dazu findet immer auch ein Vergleich mit den Modellen statt, welche auf weniger stark gefilterten Daten trainiert wurden, hier handelte es sich um einen 0.1-55Hz-Bandpassfilter.

6.2.1 Überwacht

In Tabelle 2 sind die Ergebnisse für die überwacht trainierten Encoder aufgelistet, wobei es wie gesagt zunächst um die Modelle gehen soll, bei denen die obere Grenzfrequenz des verwendeten Bandpass-filters 45Hz beträgt. Eine vollständige Auflistung der Ergebnisse inklusive der Standardabweichungen findet sich in Abschnitt B. Betrachtet man die Validierungsergebnisse, liegen diese deutlich oberhalb der Werte, die man bei einer zufälligen Verteilung erreichen würde. Die Triplet-Accuracy auf den n -back-Daten liegt mit 68% über dem mindestens zu erreichenden Zufallswert von 50%. Die Accuracy der Klassifizierung mit k -nearest-neighbours im Merkmalsraum ist mit knapp 50% auch deutlich über den 25%, die man bei einer zufälligen Zuordnung von vier Klassen erhalten würde und bestätigt somit das Ergebnis für die Triplet-Accuracy. Auch wenn hier noch Verbesserungspotenzial besteht, wirkt es so, als hätten die Modelle grundsätzlich das gelernt, was sie lernen sollten. Die mittlere Abweichung zwischen vorhergesagter und tatsächlicher Klasse mit etwa 0.6 zeigt, dass die Klassifikation im Durchschnitt weniger als eine Klasse daneben liegt. Der MAE liegt damit deutlich unterhalb des Wertes von 1.25, den man bei einer zufälligen Verteilung erhalten würde.

	n -back-TA		knn-A		knn-MAE	
Obere Grenzfrequenz [Hz]	45	55	45	55	45	55
Validierung	68.1%	80.2%	49.7%	78.0%	0.59	0.33
Test	55.1%	59.2%	30.3%	23.5%	1.13	1.29

Tabelle 2: Gemittelte Ergebnisse der überwacht trainierten Encoder auf Validierungs- und Testdaten gefiltert mit 0.1-45Hz- und 0.1-55Hz-Bandpassfilter

Auf den Testdaten zeichnet sich jedoch ein ganz anderes Bild ab. Die Triplet-Accuracy ist mit 55% kaum besser als Zufall und deutlich schlechter als die der Validierungsdaten. Gleiches gilt für die knn-Accuracy, welche bei 30% liegt. Auch die mittlere Abweichung bei den Klassifizierungen ist beim Testen fast doppelt so hoch wie bei der Validierung und damit nur noch wenig besser als Zufall. Offensichtlich gelingt es den Modellen nur sehr schlecht, auf ungewohnte EEG-Aufnahmen zu generalisieren. Denn während die Validierungsdaten aus denselben Aufnahmen stammen wie die Trainingsdaten, handelt es sich bei den Testdaten immer um eine separate Aufnahme, die das jeweilige Modell beim Training nicht gesehen hat.

Bei vergleichbaren Studien hat die Klassifikation der jeweils für die Evaluation verwendeten Daten besser funktioniert. So wurden bei einem ähnlichen n -back-Problem mit vier Klassen Klassifikations-Accuracies von 45% erzielt [18], wobei hier funktionelle Nahinfrarotspektroskopie (fNIRS) anstatt von EEG verwendet wurde. In einer anderen Studie, die EEG-Daten zur Zuordnung von n-back-Daten aus nur drei unterschiedlichen Klassen verwendete, betrug die durchschnittliche Accuracy für einen Probanden etwa 43% [19]. Ein weiteres Forschungsteam erzielte bei einer n -back-Klassifikation mit vier Klassen auf EEG-Daten sogar eine Accuracy von durchschnittlich 67% [16]. Diese Forschungsarbeiten haben im Vergleich zu dieser Arbeit etwas andere methodische Vorgehen gewählt, helfen aber bei der Einordnung der Ergebnisse. Die Accuracies auf den Validierungsdaten bewegen sich in einer ähnlichen Größenordnung wie die Klassifikationsergebnisse zweier dieser Vergleichsstudien und unterhalb der Ergebnisse von Saadati et al. [16]. Die Accuracies für die Klassifizierung der Testdaten hingegen sind deutlich niedriger. In den genannten Studien gab es jedoch keine separaten Testaufnahmen, wie es in dieser Arbeit der Fall war, da pro Proband nur eine Sitzung für die Aufnahme der Daten durchgeführt wurde. Damit unterscheiden sie sich deutlich von dieser Arbeit bezüglich der gewählten Evaluationsmethodik.

Durch Konfusionsmatrizen lässt sich die Klassifikation im Merkmalsraum der Encoder genauer untersuchen. Sie zeigen anteilig, welche Samples aus den Klassen der Ground Truth wie bei dem jeweiligen Modell klassifiziert werden. In Abbildung 16 sind die Konfusionsmatrizen für die knn-Klassifikation

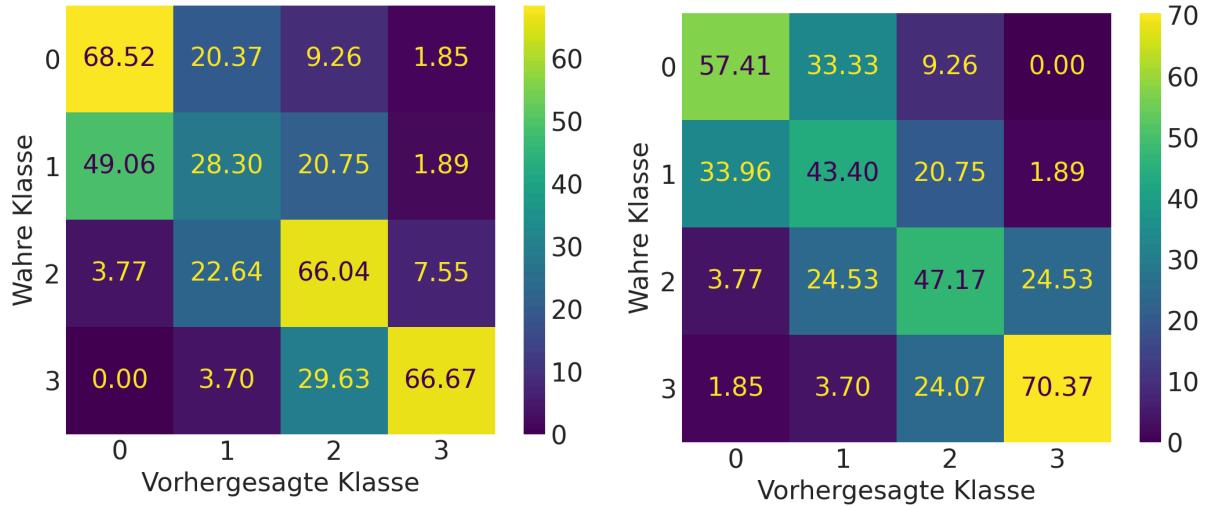


Abbildung 16: Konfusionsmatrizen für knn-Klassifikation der Validierungsdaten für zwei überwacht trainierte Encoder, 0.1-45Hz-Bandpassfilter, Testaufnahme Nr. 4

auf den Validierungsdaten für zwei überwacht trainierte Encoder zu sehen. Die Ergebnisse sehen hier aus, wie man es sich bei einer Klassifikationsaufgabe erhoffen würde. Auf der Diagonalen, die die korrekt klassifizierten Samples beschreibt, sind tendenziell die höchsten Werte in der Matrix zu finden. Fehlklassifikationen erfolgen zu großen Teilen in eine angrenzende Klasse. So gibt es beispielsweise so gut wie keine Samples aus der Klasse $n = 3$, welche in $n = 0$ einsortiert werden und anders herum. Die Klassifikation der Klassen $n = 0$ und $n = 3$, welche sich an den beiden Enden der Skala befinden, ist eher erfolgreich als die der anderen beiden. Die Ergebnisse derselben Modelle für die knn-Klassifikation der Testdaten, wie in Abbildung 17 zu sehen, schauen erkennbar anders aus. Hier sieht man, dass fast alle Samples von den beiden Modellen jeweils immer gleich klassifiziert werden. In einem Fall wird fast nur die Klasse $n = 0$ vorhergesagt, im anderen verteilen sich die Vorhersagen auf die Klassen $n = 0$ und $n = 1$. Diese Beobachtungen bestätigen die vorhergegangen Ergebnisse für Triplet-Accuracy, knn-Accuracy und MAE bezüglich der Unterschiede beim Evaluieren von Validierungs- und Testdaten. Während die Modelle Daten aus bekannten EEG-Aufnahmen recht gut einordnen können, werden Samples aus ungesehenen Aufnahmen deutlich schlechter klassifiziert, und es werden tendenziell immer dieselben ein bis zwei Klassen prognostiziert. Dies erklärt auch den hohen Wert der knn-MAE im Testfall.

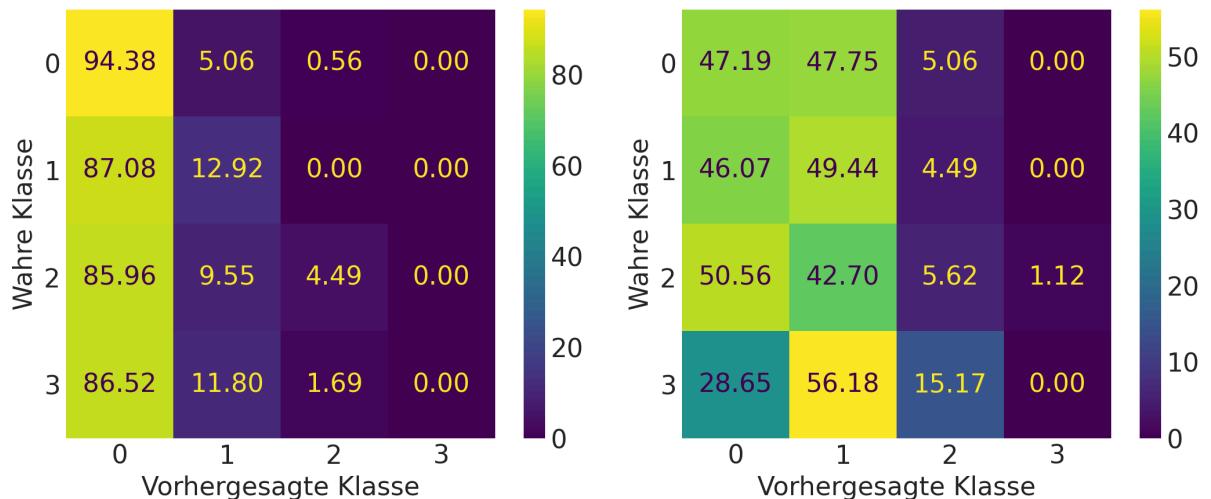


Abbildung 17: Konfusionsmatrizen für knn-Klassifikation der Testdaten für zwei überwacht trainierte Encoder, 0.1-45Hz-Bandpassfilter, Testaufnahme Nr. 4

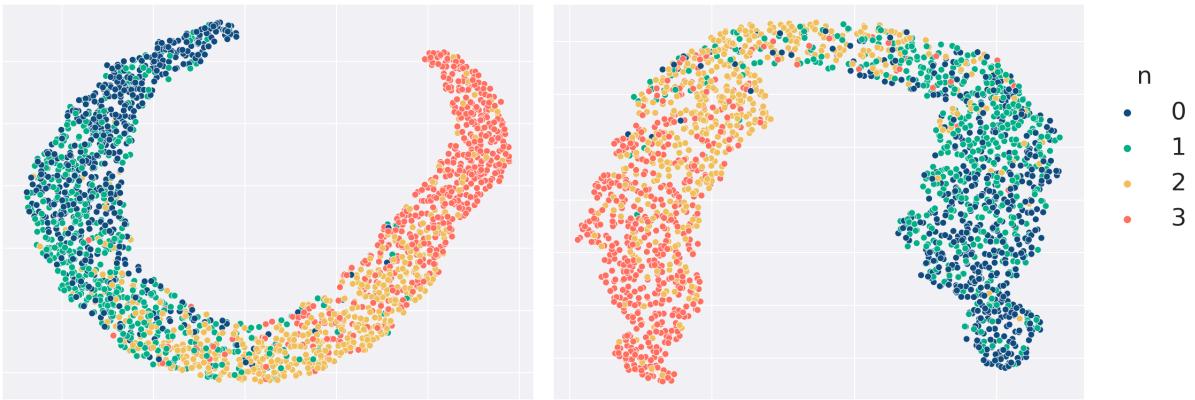


Abbildung 18: UMAP auf gelabelten Trainings- und Validierungsdaten für zwei überwacht trainierte Encoder, 0.1-45Hz-Bandpassfilter, Testaufnahme Nr. 4

Da durch das Training mit Triplet-Loss die Abbildung der Daten im Merkmalsraum beeinflusst wird, macht es Sinn, sich diesen genauer anzuschauen. Durch Dimensionsreduktion lässt sich die 64-dimensionale Ausgabe der Encoder auf zwei Dimensionen projizieren, die dann grafisch darstellbar sind. In dieser Arbeit wurde *Uniform Manifold Approximation and Projection (UMAP)* [44] zu diesem Zweck verwendet, ein nicht lineares Verfahren der Dimensionsreduktion.

In Abbildung 18 sind solche Projektionen derselben beiden überwacht trainierten Modelle zu sehen, wie schon vorherig betrachtet. Die Beschriftungen der Achsen wurden in den Darstellungen entfernt, da sie über keine nennenswerte Aussagekraft verfügen. Bei den abgebildeten Daten handelt es sich hier um die drei zum Training und zur Validierung verwendete EEG-Aufnahmen. Wie auch schon bei den vorangegangenen Evaluationsmethodiken wird deutlich, dass das Training der Modelle an sich erfolgreich war. Samples, die derselben Klasse angehören, sind tendenziell im Merkmalsraum nahe beieinander. Es sind keine voneinander getrennten Cluster für die einzelnen Klassen vorhanden, stattdessen gibt es einen fließenden Verlauf zwischen den benachbarten Klassen, eine sinnvolle Anordnung im Kontext der gestellten Klassifikationsaufgabe. Jede Klasse steht für eine Stufe auf einer Skala, die mentale Auslastung beschreiben soll. Der Übergang zwischen den Klassen von $n = 0$ über $n = 1$ und $n = 2$ hin zu $n = 3$ ist in den Darstellungen klar erkennbar.

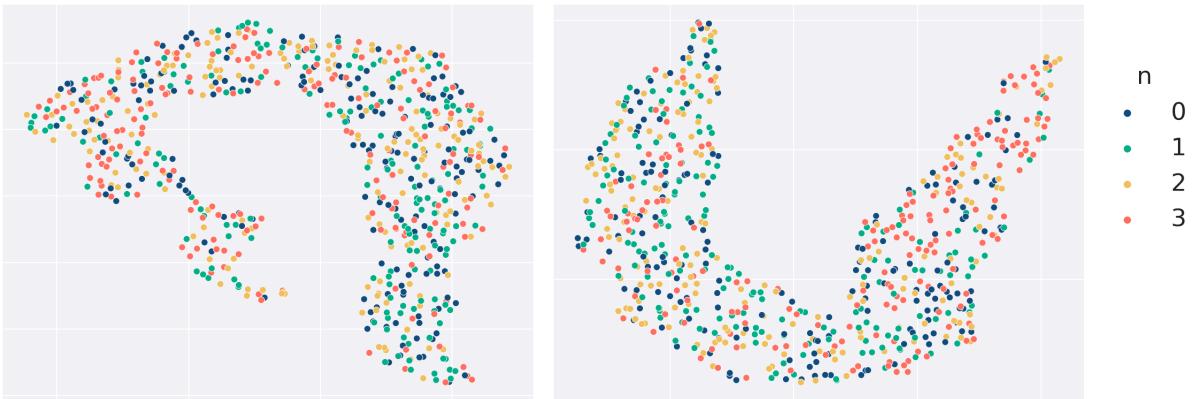


Abbildung 19: UMAP auf gelabelten Testdaten für zwei überwacht trainierte Encoder, 0.1-45Hz-Bandpassfilter, Testaufnahme Nr. 4

In Abbildung 19 ist eine UMAP-Darstellung derselben Modelle für die Testdaten zu sehen. Hier ist, anders als bei den Trainings- und Validierungsdaten, keine nennenswerte Separierung der Samples entsprechend der Klassen sichtbar. Interessant ist auch die Abbildung 20, welche die zum Training verwendeten EEG-Aufnahmen zusammen mit der Testaufnahme zeigt. Diese Darstellungen erklären die

knn-Klassifikationen der Modelle, die in den Konfusionsmatrizen aus Abbildung 17 deutlich werden. Bei dem ersten Modell wird die Testaufnahme überwiegend in dem Bereich abgebildet, in dem sich die Samples der Trainingsaufnahmen aus der Klasse $n = 0$ befinden. Folglich wurde die Testaufnahme auch zum großen Teil in diese Klasse eingesortiert. Bei dem zweiten Modell erstreckt sich die Abbildung der Testdaten über die Klassen $n = 0$ und $n = 1$, was zu der entsprechenden Klassifikation führt.

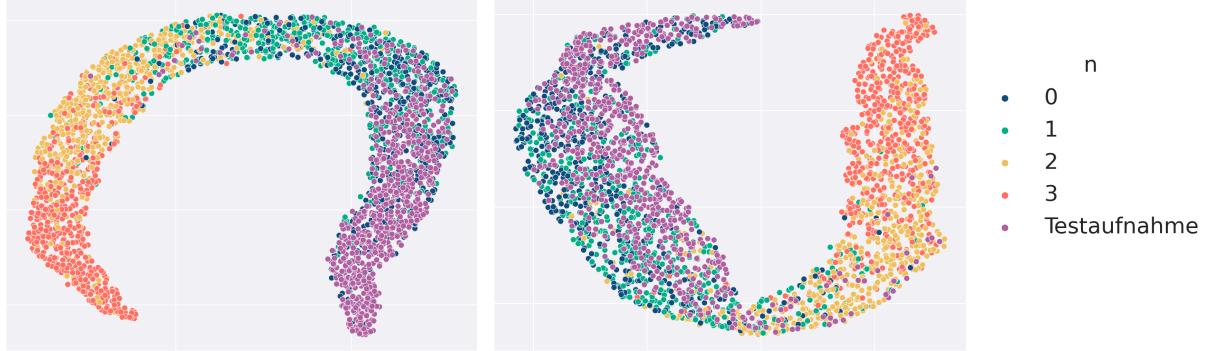


Abbildung 20: UMAP auf allen gelabelten Daten für zwei überwacht trainierte Encoder,
0.1-45Hz-Bandpassfilter, Testaufnahme Nr. 4

Nicht bei allen Encodern ist dieses Phänomen, dass die Testaufnahme im Gegensatz zu den Trainingsaufnahmen nur auf einen Teilbereich des Merkmalsraums abgebildet wird, in dieser Stärke zu beobachten. Die bisher gezeigten Modelle sind Extrembeispiele hierfür. Die Trainingsdaten stammen hier aus den ersten drei mit der n -back-Anwendung gelabelten EEG-Aufnahmen, die vierte Aufnahme wird hier zum Testen verwendet. Die knn-Accuracy der Testdaten liegt bei diesen Modellen auch nur bei ca. 27% und somit unter dem Gesamtdurchschnitt von 30%. Diese Negativbeispiele zeigen jedoch sehr deutlich, warum die Klassifikation der Testdaten grundsätzlich so viel schlechter als die der Validierungsdaten funktioniert. Anhand dieser Resultate erkennt man besonders gut, wie anders die Testaufnahme im

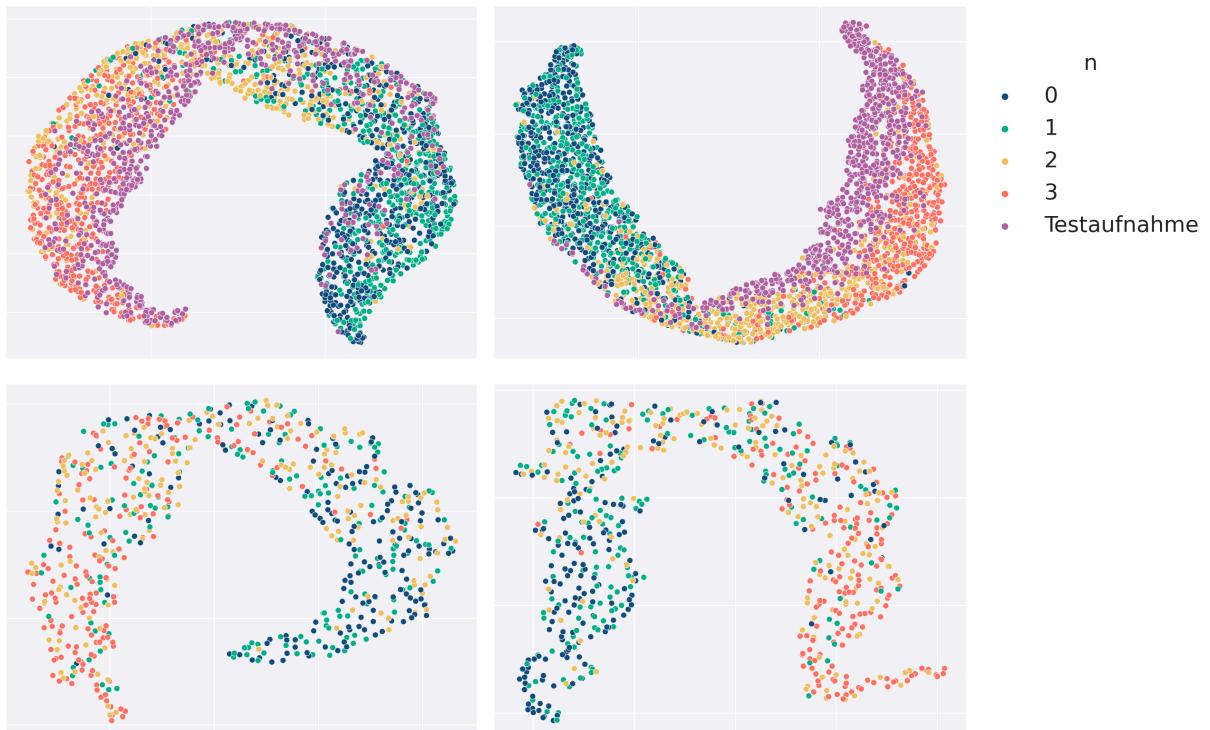


Abbildung 21: UMAP auf allen gelabelten Daten und auf Testdaten für zwei überwacht trainierte Encoder, 0.1-45Hz-Bandpassfilter, Testaufnahme Nr. 1

Vergleich zu den Trainingsaufnahmen im Merkmalsraum durch die Modelle abgebildet wird. Die besten Ergebnisse auf den Testdaten liefern die Modelle, bei denen die Aufnahme Nr. 1 zum Testen und die restlichen zum Training verwendet worden sind. Hier liegt die knn-Accuracy der Testdaten bei etwa 34%. In Abbildung 21 sind die Dimensionsreduktionen dieser Modelle dargestellt. Besonders beim ersten Modell sind die Testdaten deutlich besser über den Bereich verteilt, der auch von den Trainingsaufnahmen abgedeckt wird. Innerhalb der Testdaten ist sogar eine Anordnung der Samples entsprechend ihrer Klassen ähnlich zu der Verteilung der Trainingsdaten erkennbar. Zumindest bei den Klassen $n = 1$ und $n = 3$ ist in den Testdaten eine Separierung erkennbar.

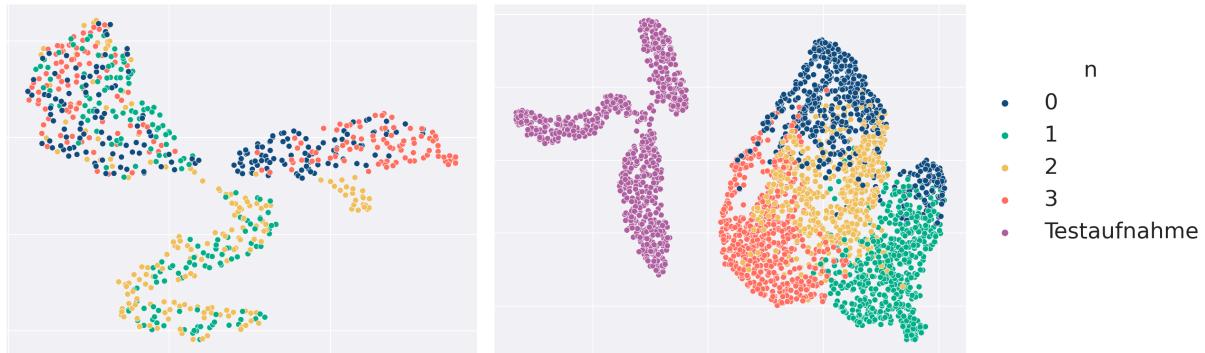


Abbildung 22: UMAP auf Testdaten und allen gelabelten Daten für einen überwacht trainierten Encoder, 0.1-55Hz-Bandpassfilter, Testaufnahme Nr. 3

Kommen wir nun zum Vergleich mit den überwacht trainierten Encodern, bei denen die EEG-Daten mit einem 0.1-55Hz-Bandpassfilter aufbereitet werden. In Tabelle 2 sieht man, dass die Ergebnisse der Modelle, bei denen die obere Grenzfrequenz bei 55Hz liegt, auf den Validierungsdaten bei allen Metriken deutlich bessere Ergebnisse erzielen als die Modelle, bei denen die Daten stärker mit einem 0.1-45Hz-Bandpassfilter gefiltert werden. Die EEG-Daten, die weniger stark gefiltert werden, scheinen somit Informationen zu enthalten, die bei dem gestellten Lernproblem hilfreich sind und in den anderen Daten fehlen. Betrachtet man die knn-Klassifikation der Testdaten, zeichnet sich jedoch ein differenzierteres Bild ab, denn hier sind die Ergebnisse im Vergleich schlechter. Mit 23% liegt die knn-Accuracy hier sogar unter dem Wert, den man bei einer zufälligen Zuordnung erhalten würde. Auch die knn-MAE ist mit 1.29 höher und somit schlechter als die der vorherig betrachteten Modelle und auch schlechter als der Wert für eine zufällige Klassifikation von 1.25. Nur die n -back-TA auf den Testdaten stellt mit 59% eine Verbesserung dar.

Diese Ergebnisse lassen sich besser nachvollziehen, wenn man die Dimensionsreduktionen des Merkmalsraums eines Encoders betrachtet, bei dem die Daten mit einem 0.1-55Hz-Bandpassfilter aufbereitet werden. In Abbildung 22 erkennt man, dass sich das Phänomen, dass die Testaufnahme anders abgebildet wird als die Trainingsaufnahmen, hier noch deutlich verstärkt. Die EEG-Aufnahme, die zum Testen verwendet wird, landet hier in einem völlig anderen Bereich als die zum Training verwendeten Aufnahmen. Folglich kann die knn-Klassifikation hier auch nicht gut funktionieren. Betrachtet man nur die Abbildung der Testdaten im Merkmalsraum, ist allerdings teilweise eine Anordnung der Samples entsprechend ihrer Klassenzugehörigkeit erkennbar. Samples aus derselben Klasse werden hier teilweise näher beianander angeordnet. Dies erklärt, warum die Triplet-Accuracy innerhalb der Testaufnahme bei diesen Modellen etwas höher liegt als bei den Modellen mit stärkerer Datenfilterung. Innerhalb der Trainingsaufnahmen sind Samples derselben Klasse sehr erfolgreich nahe beieinander liegend abgebildet. Allerdings ist hier nicht der bei den anderen Modellen erkennbare Verlauf zwischen den Klassen von $n = 0$ zu $n = 3$ entsprechend den Stufen mentaler Auslastung, für die sie stehen, erkennbar. Stattdessen gibt es sogar einen Berührungs punkt der beiden Klassen $n = 0$ und $n = 3$,

obwohl diese weit voneinander entfernt sein sollten. Auch in den Testdaten sind Samples dieser beiden Klassen in einem Bereich nahe beieinander liegend vorhanden.

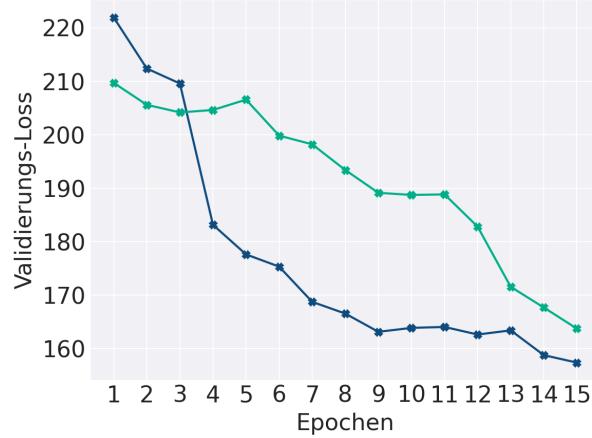


Abbildung 23: Validierungs-Loss von zwei überwacht trainierten Encodern,
0.1-45Hz-Bandpassfilter, Testaufnahme Nr. 4

Trainiert wurden die überwacht lernenden Encoder für 15 Epochen. In Abbildung 23 ist der Loss auf den Validierungsdaten zweier Modelle im Verlauf des Trainings abgebildet, das während des Trainings, so wie man es sich wünschen würde, stetig gesunken ist. Betrachtet man den Verlauf, scheint es allerdings, als hätte der Validierungs-Loss sein Minimum noch nicht erreicht und könnte bei längerem Training noch weiter sinken. Daher wurde auch längeres Training der Encoder für 50 Epochen getestet. Dies führt zwar zu einer verbesserten knn-Klassifikation der Validierungsdaten, die Ergebnisse auf der separaten Testaufnahme verschlechtern sich jedoch bei längerem Training, wie in Abbildung 24 erkennbar. Scheinbar spezialisieren sich die Modelle bei weiterem Training noch mehr auf die dafür verwendeten EEG-Aufnahmen, worunter die onehin nicht gute Generalisierung auf ungewohnte Aufnahmen zusätzlich leidet.

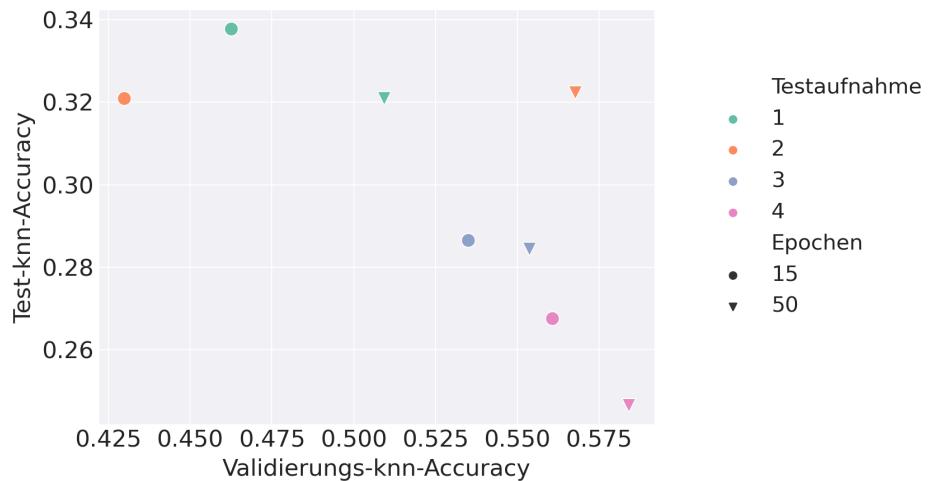


Abbildung 24: Vergleich 15 und 50 Epochen überwachtes Training, 0.1-45Hz-Bandpassfilter

6.2.2 Selbstüberwacht

Nachdem die überwacht trainierten Encoder eingehend untersucht wurden, soll es nun um die selbstüberwacht lernenden Modelle gehen. Diese wurden für 10 Epochen trainiert, in Abbildung 25 ist der Loss und die Triplet-Accuracy auf den Validierungsdaten zweier Modelle während des Trainings abgebildet. Die vollständigen gemittelten Ergebnisse mit Standardabweichungen befinden sich in Abschnitt C. Wie in Tabelle 3 erkennbar, erzielen die Modelle, bei denen die Daten mit einem

0.1-45Hz-Bandpassfilter behandelt werden, sowohl auf den Validierungs- als auch auf den Testdaten sehr hohe Triplet-Accuracies von bis zu 97%. Die Validierungsdaten sind hier eine beim Training nicht verwendete ungelabelte EEG-Aufnahme, die Testdaten die mit der n -back-Aufgabe erstellten gelabelten Daten. Je stärker die Augmentationen, desto geringer sind die Triplet-Accuracies. Dies macht Sinn, da stärkere Augmentationen das Lernproblem erschweren. Mit einem Wert von 83% für die Testdaten bei dem höchsten Augmentationsfaktor sind die Ergebnisse aber immer noch gut und deutlich über dem Wert von 50% für die Triplet-Accuracy bei einer Zufallsverteilung. Die Encoder scheinen somit erfolgreich eine Repräsentation der EEG-Daten erlernt zu haben. Hohe Triplet-Accuracies werden auch schon sehr früh im Training erreicht, wie in Abbildung 25 erkennbar. Auch gelingt es den Modellen gut, auf ungesehene EEG-Aufnahmen zu generalisieren, womit die überwacht trainierten Modelle große Probleme haben. Ein Grund hierfür könnte sein, dass die Menge an Daten und die Anzahl der beim Training verwendeten EEG-Aufnahmen im selbstüberwachten Fall deutlich größer sind. Gleichzeitig ist die Lernaufgabe hier natürlich eine andere. Für eine Klassifikation mit k -nearest-neighbours im Merkmalsraum scheinen die Repräsentationen jedoch nicht geeignet zu sein, wie die durchweg unter 25% liegende knn-Accuracy und die hohe knn-MAE zeigen. Die Klassifikationsergebnisse sind hier nicht besser als bei einer zufälligen Verteilung. Dass eine Klassifikation auf diese Weise nicht erfolgreich sein würde, ist jedoch nachvollziebar. Schließlich ist eine direkte Anwendung der Repräsentationen für die n -back-Anwendung durch die ursprüngliche selbstüberwachte Lernaufgabe in keiner Weise gegeben.

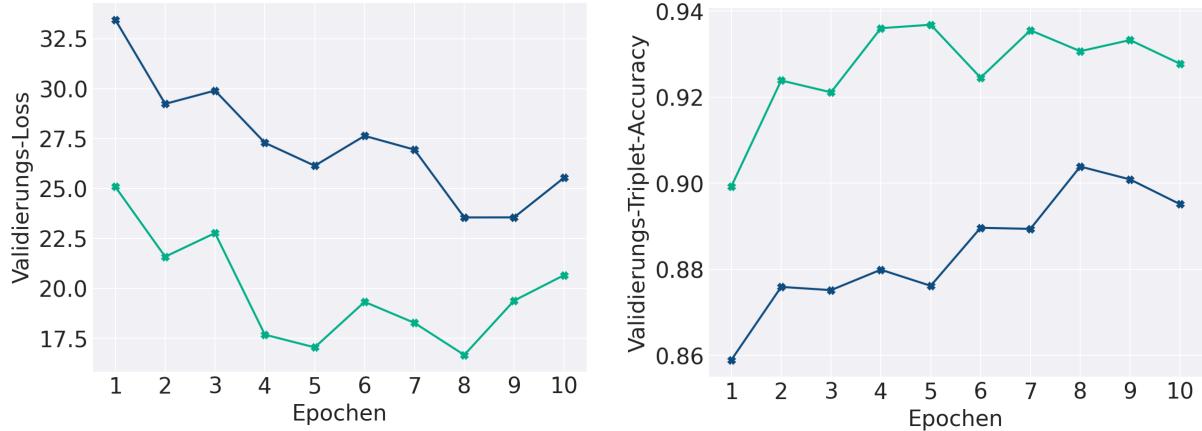


Abbildung 25: Loss und aug-TA von zwei selbstüberwacht trainierten Encodern auf Validierungsdaten, Augmentationsfaktor 1, 0.1-45Hz-Bandpassfilter

Augmentationsfaktor	Validierung		Test	
	aug-TA	aug-TA	knn-A	knn-MAE
1	96.8%	96.4%	22.9%	1.28
2	94.8%	92.8%	24.1%	1.26
3	87.9%	83.2%	24.1%	1.28

Tabelle 3: Gemittelte Ergebnisse der selbstüberwacht trainierten Encoder auf Validierungs- und Testdaten gefiltert mit 0.1-45Hz-Bandpassfilter

Betrachtet man nun in Tabelle 4 die Ergebnisse der selbstüberwacht trainierten Encoder, bei denen die Daten weniger stark mit einem 0.1-55Hz-Bandpassfilter aufbereitet werden, ist erkennbar, dass die Triplet-Accuracies besonders bei den stärkeren Augmentationen im Vergleich zu den vorher betrachteten Modellen auf den Validierungsdaten tendenziell etwas geringer sind. Vor allem fällt auf, dass die Triplet-Accuracies auf den Testdaten bei diesen Modellen für die einzelnen Augmentationsfaktoren jeweils deutlich unter den Ergebnissen für die Validierung liegen. Bei stärkerer Filterung der Daten mit

dem 0.1-45Hz-Bandpassfilter hingegen sind die Triplet-Accuracies für die ungelabelte Validierungsaufnahme und die gelabelten Testdaten recht ähnlich. Die Klassifikation mit knn hat auch bei diesen Modellen nicht gut funktioniert. Die knn-Accuracy und die knn-MAE sind nicht besser als sie bei einer zufälligen Klassifikation wären.

Augmentationsfaktor	Validierung		Test	
	aug-TA	aug-TA	knn-A	knn-MAE
1	95.7%	81.4%	24.8%	1.25
2	86.4%	67.6%	25.2%	1.22
3	80.8%	59.9%	24.9%	1.23

Tabelle 4: Gemittelte Ergebnisse der selbstüberwacht trainierten Encoder auf Validierungs- und Testdaten gefiltert mit 0.1-55Hz-Bandpassfilter

6.2.3 Klassifikationsnetz

Dass die Klassifikation im Merkmalsraum der selbstüberwacht trainierten Encoder mit knn nicht gut funktionieren würde, war erwartbar. Die erlernte Repräsentation der EEG-Daten könnte jedoch trotzdem für die anschließende Entwicklung nachgelagerter Modelle verwendbar sein. Um dies herauszufinden, wurden für ausgewählte selbstüberwacht trainierte Encoder Klassifikationsnetze trainiert, die die Ausgabe eines Encoders als Eingabe bekommen und die Klasse ausgeben. Um überwachtes und selbstüberwachtes Lernen möglichst gut vergleichen zu können, wurden solche Klassifikationsnetze auch für überwacht trainierte Encoder entwickelt. In Tabelle 5 sind die Ergebnisse der beiden Klassifikations-Methoden für die Testdaten im Vergleich dargestellt, wobei die Ergebnisse der selbstüberwachten Modelle für die drei Augmentationsfaktoren gemittelt wurden.

	knn-A	head-A	knn-MAE	head-MAE
Überwacht	30.3%	31.4%	1.16	1.17
Selbstüberwacht	23.7%	24.9%	1.27	1.26

Tabelle 5: Gemittelte Klassifikationsergebnisse ausgewählter überwacht und selbstüberwacht trainierter Encoder auf jeweiligen Testdaten gefiltert mit 0.1-45Hz-Bandpassfilter

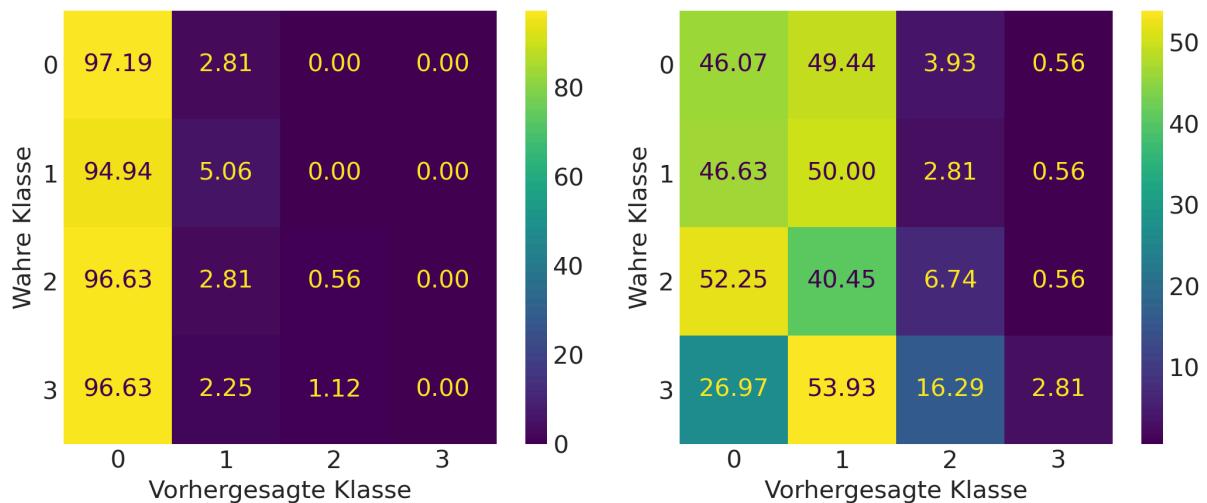


Abbildung 26: Konfusionsmatrizen für die Klassifikation der Testdaten mit neuronalem Netz für zwei überwacht trainierte Encoder, 0.1-45Hz-Bandpassfilter, Testaufnahme Nr. 4

Es ist erkennbar, dass die Klassifikation mit einem neuronalen Netz weder im überwachten noch im selbstüberwachten Fall zu einer nennenswerten Verbesserung oder Verschlechterung im Vergleich mit

der knn-Klassifikation geführt hat. Bei den überwacht trainierten Encodern ist anzunehmen, dass eine Klassifikation mit knn schon zu bestmöglichen Ergebnissen führt, da die Anwendung dieser Methodik im Kontext der Lernaufgabe unter der Verwendung von Triplet-Loss sinnvoll ist. Die Problematik der fehlenden Generalisierung auf unbekannte EEG-Aufnahmen kann durch ein Klassifikationsnetz auch nicht gelöst werden. Tatsächlich ähneln die konkreten Klassifikationen eines neuronalen Netzes sehr der knn-Klassifikation im Merkmalsraum des zugehörigen Encoders. So sind die in Abbildung 26 gezeigten Konfusionsmatrizen kaum zu unterscheiden von denen in Abbildung 17.

Die schlechten Ergebnisse der Klassifikation mit neuronalem Netz für die selbstüberwacht trainierten Encoder zeigen, dass die erlernten EEG-Repräsentationen scheinbar nicht gut für die Klassifikation mentaler Auslastung verwendbar sind. Betrachtet man die Konfusionsmatrizen in Abbildung 27, sieht man, dass die Zuordnung der Samples, ähnlich wie man es bei einer zufälligen Klassifikation erwarten würde, relativ gleichmäßig über alle Klassen verteilt erfolgt. Das Phänomen, dass die Zuordnung aller Daten mehrheitlich nur in eine oder zwei Klassen erfolgt, wie es bei den überwacht trainierten Encodern zu beobachten ist, tritt hier nicht auf.

Die Klassifikationsnetze wurden für 15 Epochen trainiert. In Abbildung 28 ist der Verlauf des Validierungs-Loss während des Trainings dargestellt, der zum Ende des Trainings zu konvergieren scheint.

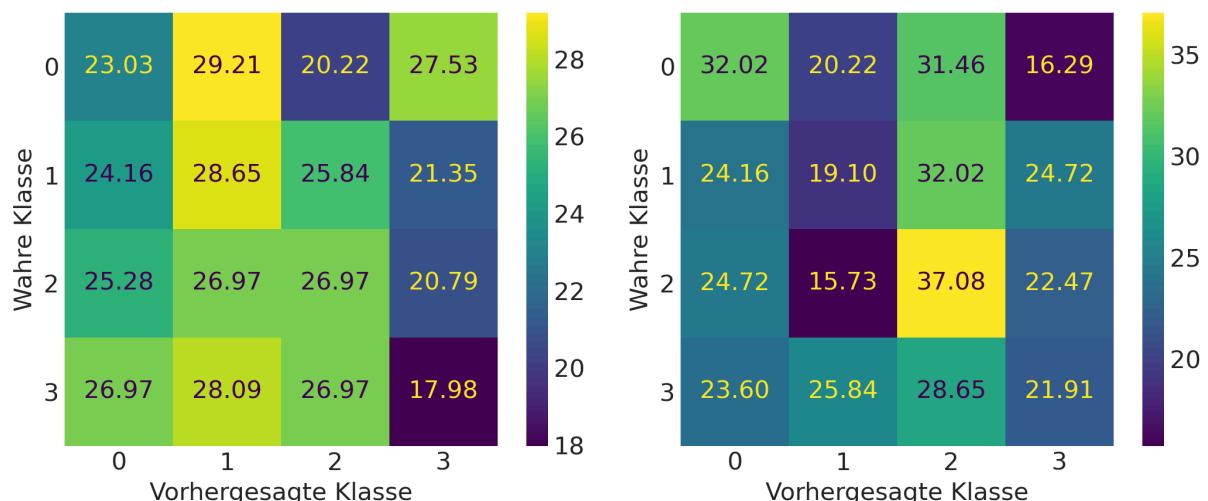


Abbildung 27: Konfusionsmatrizen für die Klassifikation der Testdaten mit neuronalem Netz für zwei selbstüberwacht trainierte Encoder, 0.1-45Hz-Bandpassfilter, Testaufnahme Nr. 4

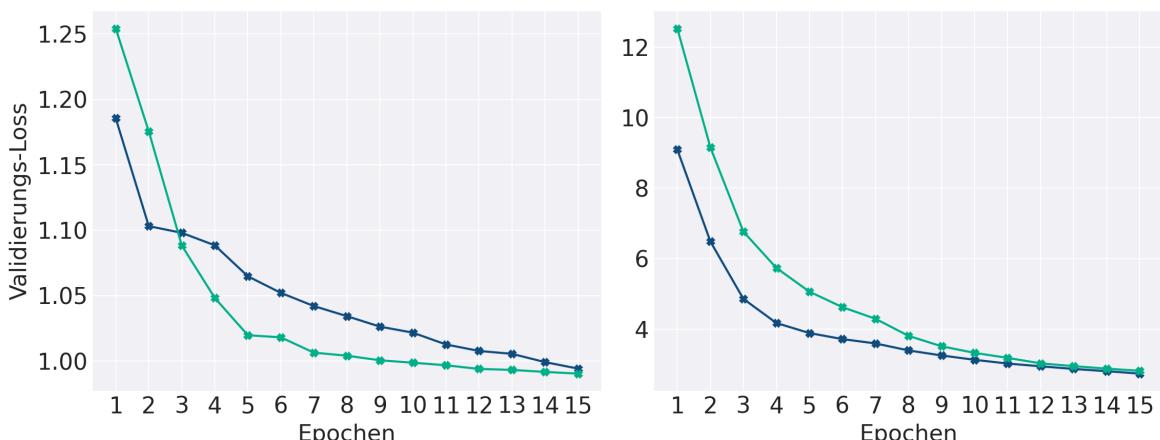


Abbildung 28: Validierungs-Loss der Klassifikationsnetze von zwei überwacht (links) und zwei selbstüberwacht (rechts) trainierten Encodern, 0.1-45Hz-Bandpassfilter

Alle betrachteten Ergebnisse für die Klassifikation mit neuronalem Netz beziehen sich auf die Modelle, bei denen die EEG-Daten mit einem 0.1-45Hz-Bandpassfilter gefiltert werden. Die Resultate der Modelle, bei denen die Daten mit einem 0.1-55Hz-Bandpassfilter aufbereitet werden, unterscheiden sich nicht nennenswert von den gezeigten Ergebnissen. Die vollständige Auflistung der Ergebnisse der trainierten Klassifizierer befindet sich in Abschnitt D.

7 Diskussion

Aufnahmespezifisches Lernen der überwacht lernenden Encoder

Ein in dieser Arbeit wiederkehrendes Thema sind die Unterschiede, die zwischen den EEG-Aufnahmen bestehen, obwohl alle Aufnahmen von derselben Person stammen. Auf den zum Training verwendeten Aufnahmen hat das überwachte Lernen bei den Encodern grundsätzlich funktioniert. Dies gilt auch für die zur Validierung verwendeten Samples, die beim Training nicht verwendet werden, aber aus den gleichen EEG-Aufnahmen stammen wie die Trainingsdaten. Die Testergebnisse auf einer separaten Aufnahme, die das Modell beim Training nicht kennengelernt hat, sind jedoch deutlich schlechter und nur noch wenig besser als Zufall. Wie man in den Dimensionsreduktionen sehen kann, wird die Testaufnahme von den Encodern anders verarbeitet als die Trainingsaufnahmen. Die überwacht trainierten Encoder spezialisieren sich beim Training also in erster Linie auf die dafür verwendeten EEG-Aufnahmen. Sie erlernen aufnahmespezifische Eigenschaften der Daten, was eine Generalisierung auf ungewohnte Aufnahmen erschwert.

Stärkere Filterung verhindert aufnahmespezifisches Lernen

Die Filterung der Daten spielt bei diesem Effekt des aufnahmespezifischen Lernens eine Rolle. So führt eine schwächere Filterung der Daten mit einem 0.1-55Hz-Bandpassfilter im Vergleich mit der stärkeren Filterung mit dem 0.1-45Hz-Bandpassfilter scheinbar vermehrt dazu, dass sich die Modelle beim Lernen auf aufnahmespezifische Merkmale stützen. Die Testaufnahme wird bei der schwächeren Filterung im Vergleich zu den zum Training verwendeten EEG-Aufnahmen in einen völlig anderen Bereich des Merkmalsraums abgebildet, und die Klassifikation dieser Aufnahme mit k -nearest-neighbours kann dadurch kaum funktionieren. Die Ergebnisse auf den Validierungsdaten sind bei der oberen Grenzfrequenz von 55Hz zwar besser, was dafür spricht, dass bei einer Filterung mit 0.1-45Hz-Bandpassfilter Informationen herausgefiltert werden, die bei der Klassifikation helfen. Diese Informationen sind jedoch kaum auf die eigentlich für diese Aufgabenstellung interessanten Hirnaktivitäten, sondern eher auf Artefakte in den Daten zurückzuführen, die beispielweise durch Muskelanspannung oder Augenbewegung entstehen können [36]. Die für die n -back-Anwendung relevanten Frequenzen

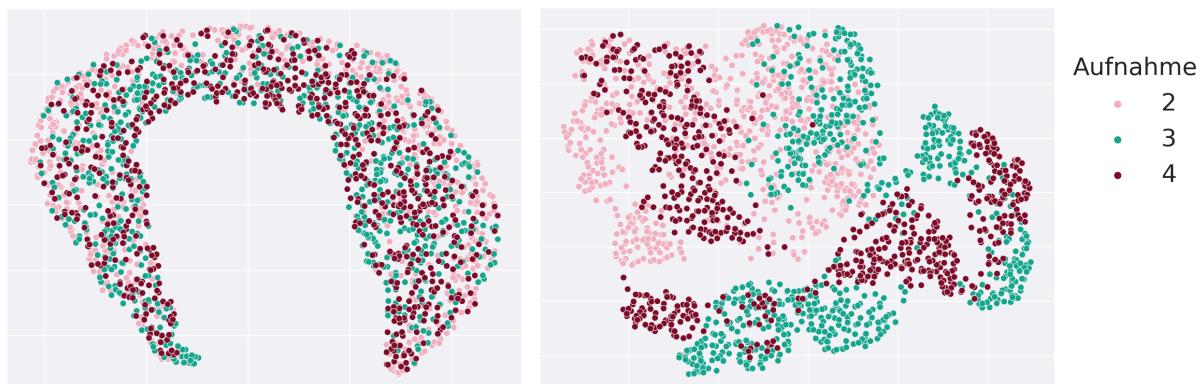


Abbildung 29: UMAP auf unterschiedlich gefilterten gelabelten Trainingsaufnahmen für zwei überwacht trainierte Encoder, 0.1-45Hz-Bandpassfilter (links), 0.1-55Hz-Bandpassfilter (rechts), Testaufnahme Nr. 1

der EEG-Daten liegen weitestgehend in einem Bereich unterhalb von 15Hz [15] und sollten somit auch bei der stärkeren Filterung mit einer Oberfrequenz von 45Hz erhalten bleiben.

Dass die Zugehörigkeit eines Samples zu der EEG-Aufnahme bei einer schwächeren Filterung eine verstärkte Rolle spielt, lässt sich auch in den Dimensionsreduktionen erkennen. Das Modell in Abbildung 29, bei dem die Trainingsdaten mit einem 0.1-55Hz-Bandpassfilter aufbereitet werden, ordnet Samples aus der gleichen Aufnahme tendenziell nahe beieinander an, was bei dem anderen Modell mit stärkerer Datenfilterung nicht passiert. Werden die Daten weniger stark gefiltert, scheinen die Encoder sich bei der Bewältigung des Lernproblems auf Eigenschaften der Daten zu stützen, die spezifisch für die einzelnen Aufnahmen sind. Dies ist vermutlich auch ein Grund dafür, dass diese Modelle gegenüber denen, bei denen mit einem 0.1-45Hz-Bandpassfilter gearbeitet wird, dafür dann keine sinnvolle Anordnung der Samples anhand der Klassen in einem Verlauf von $n = 0$ zu $n = 3$ erlernen. Denn neben der Klassenzugehörigkeit spielt hier die Aufnahme, aus der ein Sample stammt, genauso eine Rolle für die Abbildung im Merkmalsraum. Auch ein nachgelagert entwickeltes Klassifikationsnetz, das die Ausgabe eines überwacht trainierten Encoders als Eingabe erhält, kann keine bessere Generalisierung auf ungesehene EEG-Aufnahmen erreichen. Die Klassifikation mit Netz gegenüber der mit knn im Merkmalsraum des Encoders resultiert hier nicht in besseren Ergebnissen.

Stärkere Filterung fördert Angleichung von gelabelten und ungelabelten Daten

Was das selbstüberwachte Lernen angeht, haben die Encoder grundsätzlich sehr erfolgreich eine EEG-Repräsentation anhand der Augmentationen der Daten gelernt und sind auch in der Lage, auf ungesehene Aufnahmen zu generalisieren. Auch bei diesen Modellen wird jedoch deutlich, wie eine schwächere Filterung mit dem 0.1-55Hz-Bandpassfilter aufnahmespezifisches Lernen fördert. In Abbildung 30 ist zu sehen, dass die n -back-Daten, die von den selbstüberwachten Encodern nicht zum Training verwendet worden sind, bei schwächerer Filterung nur auf etwa der Hälfte des Bereichs abgebildet werden, der in der Dimensionsreduktion des Merkmalsraums von den zum Training genutzten ungelabelten EEG-Daten abgedeckt wird. Bei stärkerer Filterung sind die n -back-Daten hingegen über den ganzen Bereich verteilt. Dies erklärt auch, warum die Testergebnisse auf den n -back-Daten bei den Modellen, bei denen die Daten mit einem 0.1-55Hz-Bandpassfilter aufbereitet werden, im Vergleich zu den ungelabelten Validierungsdaten deutlich schlechter sind. Eine stärkere Filterung mit dem 0.1-45Hz-Bandpassfilter führt zu einer besseren Angleichung zwischen den gelabelten und den ungelabelten Daten. Diese Beobachtung konnte auch schon in Abschnitt 5.1 beim Vergleich der EEG-Daten nach Anwendung der beiden Filterungsarten gemacht werden. Dort konnte der 0.1-45Hz-Bandpassfilter im Vergleich zu dem 0.1-55Hz-Bandpassfilter für eine bessere Angleichung der Spektren von gelabelten und ungelabelten Daten sorgen. Ein plausibler Grund, warum sich die n -back-Daten sichtbar von den ungelabelten Daten unterscheiden, könnte sein, dass die ungelabelten Daten mehr

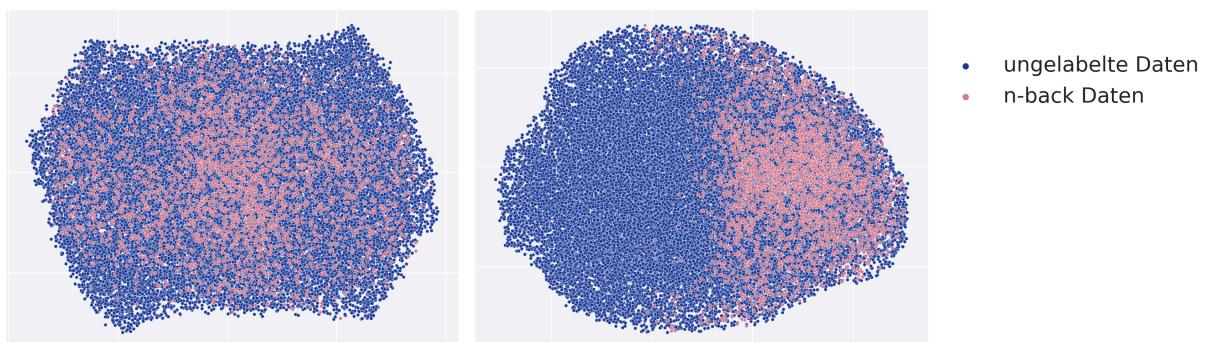


Abbildung 30: UMAP auf unterschiedlich gefilterten gelabelten und ungelabelten Daten für zwei selbstüberwacht trainierte Encoder, 0.1-45Hz-Bandpassfilter (links), 0.1-55Hz-Bandpassfilter (rechts), Augmentationsfaktor 1

Bewegungsartefakte enthalten. Während die n-back-Aufgabe innerhalb eines kurzen Zeitraums erledigt war und konzentriertes Arbeiten erforderte, wobei man sich relativ wenig bewegt, erfolgten die ungelabelten Aufnahmen über einen längeren und weniger kontrollierten Zeitraum hinweg, in dem sich mehr bewegt wurde. Diese Tatsache ließe sich sogar genauer untersuchen, da das verwendete Headset zusammen mit den EEG-Daten auch Beschleunigung und Drehbewegung gemessen hat.

Stärke der Augmentationen könnte Effekt auf aufnahmespezifisches Lernen haben

Vergleicht man die Dimensionsreduktionen der selbstüberwacht trainierten Encoder für die drei unterschiedlichen Augmentationsfaktoren, scheint es, wie in Abbildung 30 und Abbildung 31 zu sehen, dass die n-back-Daten mit verstärkenden Augmentationen zunehmend in einem mittig liegenden Teil des Bereichs der ungelabelten Daten abgebildet werden. Ob dies ein Zeichen dafür ist, dass eine stärkere Augmentation der Daten hier auch zu aufnahmespezifischerem Lernen führt, ist nicht ganz klar. Der Unterschied zwischen den Validierungsergebnissen und den Testergebnissen steigt leicht mit stärkerer Augmentation, was dafür sprechen könnte. Um dieses Phänomen genauer zu verstehen, bedarf es jedoch weiterer Untersuchungen. Hierfür ließen sich beispielsweise EEG-Daten in mehreren Sitzungen von einer Person aufnehmen, während diese ruht und keinen Aktivitäten nachgeht. Solche Daten sollten dann lediglich aufnahmespezifische Unterschiede aufweisen und könnten dann genutzt werden, um die Auswirkungen von stärkeren Augmentationen besser zu untersuchen.

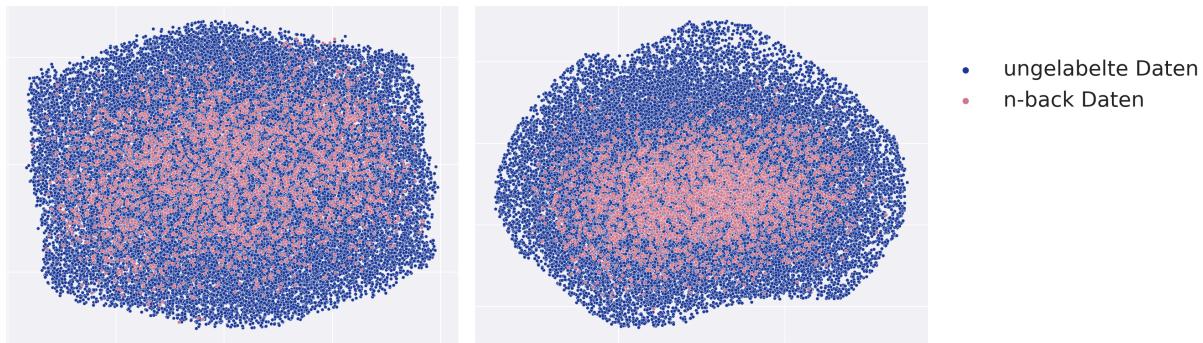


Abbildung 31: UMAP auf unterschiedlich augmentierten gelabelten und ungelabelten Daten für zwei selbstüberwacht trainierte Encoder, Augmentationsfaktor 2 (links), Augmentationsfaktor 3 (rechts), 0.1-45Hz-Bandpassfilter

0.1-45Hz-Bandpassfilter besser geeignet zur Datenaufbereitung

Aus den Ergebnissen der überwacht und selbstüberwacht trainierten Encoder lässt sich schließen, dass eine stärkere Filterung der Daten mit dem 0.1-45Hz-Bandpassfilter im Vergleich mit dem 0.1-55Hz-Bandpassfilter besser geeignet ist, um die EEG-Daten aus unterschiedlichen Aufnahmen einander anzugeleichen und Artefakte in den Daten zu entfernen. Mit der stärkeren Filterung lernen die Modelle eher die eigentlich interessanten Merkmale der Daten über Aufnahmen hinweg, anstatt sich auf aufnahmespezifische Eigenschaften zu konzentrieren. Trotzdem hat die Generalisierung auch mit 0.1-45Hz-Bandpassfilter auf ungesehene EEG-Aufnahmen im überwachten Fall nicht sonderlich gut funktioniert, bei den selbstüberwacht lernenden Modellen hingegen schon. Dies wird unter anderem sicherlich an der geringen Anzahl an n-back-Daten gegeben haben. Nur drei EEG-Aufnahmen von einer Gesamtlänge von etwa 40 Minuten zum Training der überwacht lernenden Encoder zu verwenden wird angesichts des prominenten Problems der Unterschiedlichkeit einzelner Aufnahmen zu wenig gewesen sein. Im selbstüberwachten Fall wurden die Modelle auf acht EEG-Aufnahmen trainiert, die sich zusammen auf etwa neun Stunden Datenaufnahme summieren. Ständen mehr gelabelte Aufnahmen zur Verfügung, wäre es dann auch möglich und angebracht, ebenso beim überwachten Lernen eine eigene Aufnahme für die Validierung zu verwenden. Des Weiteren wäre es in diesem Zusammenhang

natürlich auch sinnvoll, EEG-Daten von nicht nur einer, sondern von unterschiedlichen Personen zu verwenden, was im Rahmen dieser Arbeit leider nicht möglich war.

Variierende Aufnahmebedingungen fördern Unterschiede in den Daten

Eine weitere Ursache für die starken Differenzen zwischen den EEG-Aufnahmen werden die Aufnahmebedingungen gewesen sein. Die Aufnahmen wurden hier in einem relativ alltagsnahen Setting gemacht, wobei keine Nass-, sondern Trocken-Elektroden verwendet wurden, die im Vergleich zu einer geringeren Signalqualität führen. Allein die Elektrodenplatzierung wird bei jeder Aufnahme leicht variiert haben. Auf eine Bereinigung der Daten wurde, abgesehen von der Bandpass-Filterung, weitestgehend verzichtet. Es sollte in dieser Arbeit eben auch untersucht werden, wie viel sich mit solchen alltagsnah aufgenommenen EEG-Daten anfangen lässt. Es ist vorstellbar, dass diese Aufnahmebedingungen unter anderem dazu beigetragen haben, dass sich die EEG-Aufnahmen relativ stark voneinander unterscheiden. Wie man EEG-Aufnahmen für die Verwendung beim maschinellen Lernen aneinander angleicht und dem aufnahmespezifischen Lernen besser entgegenwirken kann, ist ein Bereich, der noch weiter erforscht werden sollte. So könnte man zum Beispiel noch stärkere Filterungen der Daten testen oder untersuchen, ob andere kontrastive oder generative Lerntechniken vielleicht weniger anfällig dafür sind, dass die Modelle aufnahmespezifische Eigenschaften lernen. Auch Verfahren der Merkmalsextraktion aus EEG-Daten könnten in dieser Hinsicht untersucht werden. Inwieweit die konkreten Aufnahmebedingungen zu dieser Problematik beitragen, ist ebenso interessant. War das Phänomen der Unterschiedlichkeit der EEG-Aufnahmen in dieser Arbeit sehr präsent, taucht dies, so weit ich das feststellen konnte, in der relevanten Literatur kaum auf. Ein Mangel von Forschungsarbeiten in diesem Bereich ist jedoch, dass anders als in dieser Arbeit pro Person oft nur eine Sitzung für die Datenaufnahme erfolgt. Inwieweit die entwickelten Klassifizierer, die häufig spezifisch auf einen Probanden ausgelegt sind, dann auf neue Aufnahmen dieser Person generalisieren können, ist dann folglich nicht untersuchbar. Gerade hinsichtlich einer potentiellen Verwendung in Brain-Computer-Interfaces ist dies jedoch eine relevante Evaluationsmethodik. Die schlechten Testergebnisse der für diese Arbeit überwacht entwickelten Encoder lassen sich so auch schlecht mit den scheinbar besseren Resultaten aus anderen Studien vergleichen.

Selbstüberwacht erlernte Repräsentationen nicht für n-back-Klassifikation verwendbar

Die selbstüberwacht trainierten Encoder haben erfolgreich EEG-Repräsentationen der Daten anhand der beim Training verwendeten Augmentationen gelernt. Auch auf den beim Training ungewohnten Aufnahmen erzielen die Encoder ähnlich gute Ergebnisse wie auf den Trainingsaufnahmen. Die erlernten Repräsentationen lassen sich jedoch nicht gut weiterverwenden, um sie für eine Klassifikation mentaler Auslastung im Rahmen der *n-back*-Aufgabe zu nutzen. In der Studie, die als Vorbild für das kontrastive selbstüberwachte Lernen mit augmentierten EEG-Daten diente [14], hat eine ähnliche Methodik wiederum gut funktioniert, wobei die Umstände dieser Studie sich in einigen Punkten von denen dieser Arbeit unterscheiden. Ein wichtiger Unterschied ist, dass die Merkmalsextraktion aus den Daten in der Vorbild-Studie jeweils für die einzelnen EEG-Kanäle anstatt für alle Kanäle zusammen erfolgte. Es wurden andere EEG-Daten verwendet, und die nachgelagerten Klassifikationsaufgaben, welche zur Evaluation der angewandten Methodik des selbstüberwachten Lernens genutzt wurden, bewegten sich nicht im Bereich mentaler Auslastung. Darüber hinaus wurde mit der Frequenzverschiebung in dieser Arbeit eine zusätzliche Datenaugmentation verwendet. Im konkreten Vorgehen gibt es noch weitere Unterschiede. Woran es genau liegt, dass diese Methodik des kontrastiven selbstüberwachten Lernens bei Mohsenvand et al. [14] erfolgreich war und in dieser Arbeit nicht, lässt sich jedoch nicht genau sagen und müsste genauer untersucht werden. Die in der Vorbildstudie verwendeten Datensätze sind frei zugänglich, die von den Forschenden verwendete Implementierung jedoch nicht. Mit diesen Datensätzen ließe sich jedoch zum Beispiel das in dieser Abschlussarbeit verwendete Vorgehen wiederholen. Sollten die selbstüberwacht erlernten Repräsentationen auch mit diesen

Daten nicht verwertbar sein, sind die unterschiedlichen Ergebnisse auf Differenzen im methodischen Vorgehen zurückzuführen und liegen nicht oder nicht allein an den jeweils verwendeten Datensätzen. Interessant wäre auch anders herum die genannte Forschungsarbeit mit den hier verwendeten *n*-back-Daten nachzuahmen.

Energieverbrauch beim Training

Da die Nutzung von Anwendungen des maschinellen Lernens und insbesondere die Entwicklung großer Modelle mit vielen Parametern, die lange und auf großen Datenmengen trainiert werden, zunimmt, treten in diesem Bereich verstärkt auch Nachhaltigkeitsaspekte in den Vordergrund. Bei der Entwicklung und der anschließenden Nutzung von solchen Modellen werden teilweise hohe Mengen an Energie verbraucht. *Green AI* ist daher ein Forschungsbereich mit steigender Relevanz [45]–[47]. In dieser Arbeit liegt kein großer Fokus auf dem Thema Nachhaltigkeit, aber es wurde beim Training aller im Rahmen dieses Projekts entwickelten Modelle nebenbei der Energieverbrauch aufgenommen. Insgesamt wurden für diese Abschlussarbeit 479 Modelle trainiert. Die Messungen wurden mithilfe von *codecarbon* [48] durchgeführt. Diese Bibliothek misst die Energie, die von RAM, Prozessor und GPU verbraucht wird. In der Summe wurden so für das Training 92.17 kWh aufgewendet, was dem Energieverbrauch eines durchschnittlichen deutschen Haushalts im Jahr 2021⁷ über fast zwei Tage (45.23 Stunden) entspricht. Das Training eines einzelnen Transformers [49] mit 65 Mio. Parametern auf acht NVIDIA P100 GPUs für 12 Stunden benötigt zum Vergleich etwa 17 kWh [47].

8 Fazit

In dieser Arbeit wurde ein Ansatz des kontrastiven selbstüberwachten Lernens für die Klassifikation von mentaler Auslastung auf EEG-Daten exploriert. Gleichzeitig wurde eine ähnliche Methodik für die Entwicklung von überwacht lernenden Modellen verwendet. Die Architektur der überwacht und selbstüberwacht trainierten Encoder beruhte dabei auf dem EEGNet [20], einem kompakten CNN. Für das Training der Encoder wurden zwei Datensätze erstellt. Der gelabelte Datensatz umfasste vier Stufen mentaler Auslastung, die mit der *n*-back-Aufgabe induziert wurden. Darüber hinaus wurde eine größere Menge an ungelabelten EEG-Daten aufgenommen, die beim selbstüberwachten Lernen Anwendung fanden. Hier wurden Augmentationen der ungelabelten Daten durchgeführt, um unter der Verwendung von Triplet-Loss Encoder zu trainieren, die Repräsentationen der EEG-Daten erlernten. Das Training der selbstüberwacht lernenden Encoder war dabei zwar erfolgreich, die erlernten Repräsentationen konnten von einem im Anschluss entwickelten Netz, das als Klassifizierer diente, jedoch nicht genutzt werden, um eine korrekte Zuordnung der *n*-back-Daten zu den Stufen mentaler Auslastung durchzuführen. Der hier gewählte selbstüberwachte Ansatz war angelehnt an eine Studie, in der gute Ergebnisse mit einer ähnlichen Methodik erzielt wurden [14]. Warum er in dieser Arbeit nicht erfolgreich war, ist unklar und bedarf weiterer Untersuchungen. Auf den *n*-back-Daten wurden überwacht lernende Encoder trainiert, wobei auch hier Triplet-Loss Anwendung fand. Die überwacht erlernten Repräsentationen konnten einerseits mithilfe von knn im Merkmalsraum klassifiziert werden. Des Weiteren wurden auch hier im Anschluss Klassifizierer entwickelt, welche jedoch ähnliche Ergebnisse erzielten wie die Klassifikation mit knn. Bei EEG-Daten, die aus den gleichen Aufnahmen wie die zum Training verwendeten Daten stammten, gelang die Klassifikation recht gut. Bei einer Aufbereitung der Daten mit dem 0.1-45Hz-Bandpassfilter wurden hier Accuracies von etwa 50% erzielt. Auch die Anordnung der Samples im Merkmalsraum war in Anbetracht der gestellten Klassifikationsaufgabe sinnvoll. Wurden jedoch Daten aus einer beim Training nicht verwendeten EEG-Aufnahme verwendet, waren die Ergebnisse nur noch wenig besser als Zufall, die Accuracy lag hier bei etwa 31%. Die überwacht trainierten Encoder scheinen sich in ihrem Lernen relativ stark auf Eigenschaften

⁷<https://www.destatis.de/EN/Themes/Society-Environment/Environment/Environmental-Economic-Accounting/private-households/Tables/energy-consumption-households.html>

zu stützen, die spezifisch für die jeweilige Aufnahme sind, aus der ein Sample stammt, wodurch eine Generalisierung auf ungewohnte Aufnahmen schwer fällt. Aufnahmespezifischem Lernen konnte hier teilweise durch eine Filterung der EEG-Daten entgegengewirkt werden. Dabei hat ein 0.1-45Hz-Bandpassfilter zur Aufbereitung der Daten im Vergleich mit einem schwächeren 0.1-55Hz-Bandpassfilter eher zu einer Angleichung der EEG-Aufnahmen geführt. Auch hat die Verwendung des stärkeren 0.1-45Hz-Bandpassfilters sichtbare Unterschiede zwischen den gelabelten und ungelabelten Daten weitestgehend beseitigen können. Die überwacht trainierten Encoder haben jedoch trotz stärkerer Filterung der Daten auf einer separaten beim Training nicht verwendeten Testaufnahme keine guten Ergebnisse erzielen können. Dieses Phänomen des aufnahmespezifischen Lernens bei der Verwendung von EEG-Daten und wie es sich vermeiden ließe ist ebenso ein Bereich, der weiter erforscht werden sollte.

Referenzen

- [1] L. F. Nicolas-Alonso und J. Gomez-Gil, „Brain Computer Interfaces, a Review“, *Sensors*, Bd. 12, Nr. 2, S. 1211–1279, 2012, doi: 10.3390/s120201211.
- [2] M. F. Mridha, S. C. Das, M. M. Kabir, A. A. Lima, M. R. Islam, und Y. Watanobe, „Brain-Computer Interface: Advancement and Challenges“, *Sensors*, Bd. 21, Nr. 17, 2021, doi: 10.3390/s21175746.
- [3] J. J. Shih, D. J. Krusienski, und J. R. Wolpaw, „Brain-Computer Interfaces in Medicine“, *Mayo Clinic Proceedings*, Bd. 87, Nr. 3, S. 268–279, 2012, doi: 10.1016/j.mayocp.2011.12.008.
- [4] C.-Z. Ou, B.-S. Lin, C.-J. Chang, und C.-T. Lin, „Brain Computer Interface-based Smart Environmental Control System“, in *2012 Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 2012, S. 281–284. doi: 10.1109/IIH-MSP.2012.74.
- [5] G. A. M. Vasiljevic und L. C. de Miranda, „Brain–Computer Interface Games Based on Consumer-Grade EEG Devices: A Systematic Literature Review“, *International Journal of Human–Computer Interaction*, Bd. 36, Nr. 2, S. 105–142, 2020, doi: 10.1080/10447318.2019.1612213.
- [6] B. F. Yuksel *u. a.*, „Learn Piano with BACH: An Adaptive Learning Interface that Adjusts Task Difficulty Based on Brain State“, in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, in CHI '16. San Jose, California, USA: Association for Computing Machinery, 2016, S. 5372–5384. doi: 10.1145/2858036.2858388.
- [7] M. Hajinorozi, Z. Mao, T.-P. Jung, C.-T. Lin, und Y. Huang, „EEG-based prediction of driver's cognitive performance by deep convolutional neural network“, *Signal Processing: Image Communication*, Bd. 47, S. 549–555, 2016, doi: 10.1016/j.image.2016.05.018.
- [8] F. Dehais, R. Roy, und S. Scannella, „Inattentional deafness to auditory alarms: Inter-individual differences, electrophysiological signature and single trial classification“, *Behavioural Brain Research*, Bd. 360, S. 51–59, 2019, doi: 10.1016/j.bbr.2018.11.045.
- [9] K. M. Hossain, M. A. Islam, S. Hossain, A. Nijholt, und M. A. R. Ahad, „Status of deep learning for EEG-based brain–computer interface applications“, *Frontiers in Computational Neuroscience*, Bd. 16, 2023, doi: 10.3389/fncom.2022.1006763.
- [10] M. H. Rafiei, L. V. Gauthier, H. Adeli, und D. Takabi, „Self-Supervised Learning for Electroencephalography“, *IEEE Transactions on Neural Networks and Learning Systems*, Bd. 35, Nr. 2, S. 1457–1471, 2024, doi: 10.1109/TNNLS.2022.3190448.
- [11] J. Xu, Y. Zheng, Y. Mao, R. Wang, und W.-S. Zheng, „Anomaly Detection on Electroencephalography with Self-supervised Learning“, in *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2020, S. 363–368. doi: 10.1109/BIBM49941.2020.9313163.
- [12] C. Yang, D. Xiao, M. B. Westover, und J. Sun, „Self-supervised EEG Representation Learning for Automatic Sleep Staging“, *arXiv e-prints*, S. arXiv:2110.15278, Okt. 2021, doi: 10.48550/arXiv.2110.15278.
- [13] H. J. Banville, I. Albuquerque, A. Hyvärinen, G. Moffat, D.-A. Engemann, und A. Gramfort, „Self-supervised representation learning from electroencephalography signals“, *CoRR*, 2019, doi: 10.48550/arXiv.1911.05419.
- [14] M. N. Mohsenvand, M. R. Izadi, und P. Maes, „Contrastive Representation Learning for Electroencephalogram Classification“, in *Proceedings of the Machine Learning for Health NeurIPS Workshop*, E. Alsentzer, M. B. A. McDermott, F. Falck, S. K. Sarkar, S. Roy, und S. L. Hyland, Hrsg., in *Proceedings of Machine Learning Research*, vol. 136. PMLR, 2020, S. 238–253. [Online]. Verfügbare unter: <https://proceedings.mlr.press/v136/mohsenvand20a.html>

- [15] A.-M. Brouwer, M. Hogervorst, J. Erp, T. Heffelaar, P. Zimmerman, und R. Oostenveld, „Estimating workload using EEG spectral power and ERPs in the n-back task“, *Journal of neural engineering*, Bd. 9, S. 45008–45009, 2012, doi: 10.1088/1741-2560/9/4/045008.
- [16] M. Saadati, J. Nelson, und H. Ayaz, „Multimodal fNIRS-EEG Classification Using Deep Learning Algorithms for Brain-Computer Interfaces Purposes“, in *Advances in Neuroergonomics and Cognitive Engineering*, H. Ayaz, Hrsg., Cham: Springer International Publishing, 2020, S. 209–220. doi: 10.1007/978-3-030-20473-0_21.
- [17] D. Grimes, D. S. Tan, S. E. Hudson, P. Shenoy, und R. P. Rao, „Feasibility and pragmatics of classifying working memory load with an electroencephalograph“, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, in CHI '08. Florence, Italy: Association for Computing Machinery, 2008, S. 835–844. doi: 10.1145/1357054.1357187.
- [18] C. Herff, D. Heger, O. Fortmann, J. Hennrich, F. Putze, und T. Schultz, „Mental workload during n-back task—quantified in the prefrontal cortex using fNIRS“, *Frontiers in Human Neuroscience*, Bd. 7, 2014, doi: 10.3389/fnhum.2013.00935.
- [19] Y. Liu, H. Ayaz, und P. A. Shewokis, „Mental workload classification with concurrent electroencephalography and functional near-infrared spectroscopy“, *Brain-Computer Interfaces*, Bd. 4, Nr. 3, S. 175–185, 2017, doi: 10.1080/2326263X.2017.1304020.
- [20] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, und B. J. Lance, „EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces“, *Journal of Neural Engineering*, Bd. 15, Nr. 5, S. 56013–56014, Juli 2018, doi: 10.1088/1741-2552/aace8c.
- [21] S. Zschocke und H.-C. Hansen, *Klinische Elektroenzephalographie*. in Springer eBook Collection, Medicine. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. doi: 10.1007/978-3-662-08106-8.
- [22] A. Jung, *Maschinelles Lernen - Die Grundlagen*, 1st ed. 2024. Singapore: Springer Nature Singapore, 2024. doi: 10.1007/978-981-99-7972-1.
- [23] I. Goodfellow, Y. Bengio, und A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Verfügbar unter: <http://www.deeplearningbook.org/>
- [24] X. Liu *u. a.*, „Self-Supervised Learning: Generative or Contrastive“, *IEEE Transactions on Knowledge and Data Engineering*, Bd. 35, Nr. 1, S. 857–876, 2023, doi: 10.1109/TKDE.2021.3090866.
- [25] C. Borgelt, F. Klawonn, C. Moewes, G. Ruß, M. Steinbrecher, und R. Kruse, *Computational Intelligence - Eine methodische Einführung in Künstliche Neuronale Netze, Evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze*. in Studium. Wiesbaden: Vieweg + Teubner, 2012. doi: 10.1007/978-3-8348-8299-8.
- [26] Z. Liu, A. Alavi, M. Li, und X. Zhang, „Self-Supervised Contrastive Learning for Medical Time Series: A Systematic Review“, *Sensors*, Bd. 23, Nr. 9, 2023, doi: 10.3390/s23094221.
- [27] F. D. Pup und M. Atzori, „Applications of Self-Supervised Learning to Biomedical Signals: A Survey“, *IEEE Access*, Bd. 11, Nr. , S. 144180–144203, 2023, doi: 10.1109/ACCESS.2023.3344531.
- [28] T. Chen, S. Kornblith, M. Norouzi, und G. E. Hinton, „A Simple Framework for Contrastive Learning of Visual Representations“, *CoRR*, 2020, doi: 2002.05709.
- [29] L. Cabañero Gómez, R. Hervás, I. González, und V. Villarreal, „Studying the generalisability of cognitive load measured with EEG“, *Biomedical Signal Processing and Control*, Bd. 70, S. 103032–103033, 2021, doi: 10.1016/j.bspc.2021.103032.
- [30] T. Stenner *u. a.*, „sccn/libsl: v1.16.0“. März 2022. doi: 10.5281/zenodo.6387090.

- [31] T. Stenner *u. a.*, „LabRecorder“. [Online]. Verfügbar unter: <https://github.com/labstreaminglayer/App-LabRecorder/releases/tag/v1.16.0>
- [32] S. Williams *u. a.*, „Eel“. [Online]. Verfügbar unter: <https://github.com/python-eel/Eel/releases/tag/v0.15.0>
- [33] C. Boulay *u. a.*, „PyLSL“. [Online]. Verfügbar unter: <https://github.com/labstreaminglayer/pylsl/releases/tag/v1.16.0>
- [34] W. McKinney, „Data Structures for Statistical Computing in Python“, in *Proceedings of the 9th Python in Science Conference*, S. van der Walt und J. Millman, Hrsg., 2010, S. 56–61. doi: 10.25080/Majora-92bf1922-00a .
- [35] E. Larson *u. a.*, „MNE-Python“. November 2022. doi: 10.5281/zenodo.7314185.
- [36] W. O. Tatum, B. A. Dworetzky, und D. L. Schomer, „Artifact and Recording Concepts in EEG“, *Journal of Clinical Neurophysiology*, Bd. 28, Nr. 3, S. 252–263, 2011.
- [37] L. Brabetz, O. Haas, und C. Koppe, *Grundgebiete der Elektrotechnik Band 2: Wechselströme, Drehstrom, Leitungen, Anwendungen der Fourier-, der Laplace- und der Z-Transformation*. Berlin, Boston: De Gruyter Oldenbourg, 2023. doi: 10.1515/9783110631647.
- [38] G. Montavon, G. B. Orr, und K.-R. Müller, Hrsg., *Neural Networks: Tricks of the Trade: Second Edition*, Bd. 7700. in Lecture Notes in Computer Science, vol. 7700. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. doi: 10.1007/978-3-642-35289-8.
- [39] F. Pedregosa *u. a.*, „Scikit-learn: Machine Learning in Python“, *Journal of Machine Learning Research*, Bd. 12, S. 2825–2830, 2011.
- [40] A. Paszke *u. a.*, „PyTorch: An Imperative Style, High-Performance Deep Learning Library“, in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, und R. Garnett, Hrsg., Curran Associates, Inc., 2019, S. 8024–8035. [Online]. Verfügbar unter: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>
- [41] V. Godbole, G. E. Dahl, J. Gilmer, C. J. Shallue, und Z. Nado, „Deep Learning Tuning Playbook“. [Online]. Verfügbar unter: http://github.com/google-research/tuning_playbook
- [42] C. Rommel, J. Paillard, T. Moreau, und A. Gramfort, „Data augmentation for learning predictive models on EEG: a systematic comparison“, *Journal of Neural Engineering*, Bd. 19, Nr. 6, S. 66020–66021, Nov. 2022, doi: 10.1088/1741-2552/aca220.
- [43] P. Virtanen *u. a.*, „SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python“, *Nature Methods*, Bd. 17, S. 261–272, 2020, doi: 10.1038/s41592-019-0686-2.
- [44] L. McInnes, J. Healy, N. Saul, und L. Grossberger, „UMAP: Uniform Manifold Approximation and Projection“, *The Journal of Open Source Software*, Bd. 3, Nr. 29, S. 861–862, 2018.
- [45] N. Alizadeh und F. C. Filho, „Green AI: A Preliminary Empirical Study on Energy Consumption in DL Models Across Different Runtime Infrastructures“, *ArXiv*, 2024, [Online]. Verfügbar unter: <https://api.semanticscholar.org/CorpusID:267770026>
- [46] A. Lacoste, A. Lucioni, V. Schmidt, und T. Dandres, „Quantifying the Carbon Emissions of Machine Learning“. 2019.
- [47] E. Strubell, A. Ganesh, und A. McCallum, „Energy and Policy Considerations for Deep Learning in NLP“. 2019.

- [48] V. Schmidt *u. a.*, „mlco2/codecarbon: v2.1.4“. [Online]. Verfügbar unter: <https://doi.org/10.5281/zenodo.7049269>
- [49] A. Vaswani *u. a.*, „Attention is All you Need“, in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, und R. Garnett, Hrsg., Curran Associates, Inc., 2017, S. . [Online]. Verfügbar unter: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fdb053c1c4a845aa-Paper.pdf

Anhang

A) Angepasste EEGNet Architektur

Block	Schicht	# Filter	Größe	Ausgabe	Aktivierung	Optionen
Eingabe				(8, 500)		
1	Conv2D	8	(1, 125)	(8, 8, 500)	Linear	padding=same
	BatchNorm			(8, 8, 500)		
	Depthwise	2 · 8	(8, 1)	(16, 1, 500)	Linear	padding=valid, depth=2
	Conv2D					
	BatchNorm			(16, 1, 500)		
	Activation			(16, 1, 500)	ELU	
	AveragePool2D		(1, 4)	(16, 1, 125)		
	Dropout			(16, 1, 125)		$p = 0.5$
	Separable Conv2D	16	(1, 31)	(16, 1, 125)		padding=same
	BatchNorm			(16, 1, 125)		
2	Activation			(16, 1, 125)	ELU	
	AveragePool2D		(1, 8)	(16, 1, 15)		
	Dropout			(16, 1, 15)		$p = 0.5$
	Flatten			(240)		
	FullyConnected	64		(64)		
Ausgabe						

B) Ergebnisse der überwacht trainierten Encoder

B.1) 0.1-45Hz-Bandpassfilter

	<i>n</i> -back-TA		knn-A		knn-MAE	
	Mittel	SD	Mittel	SD	Mittel	SD
Validierung	68.1%	2.8%	49.7%	6.1%	0.59	0.08
Test	55.1%	4.3%	30.3%	4.0%	1.13	0.15

B.2) 0.1-55Hz-Bandpassfilter

	<i>n</i> -back-TA		knn-A		knn-MAE	
	Mittel	SD	Mittel	SD	Mittel	SD
Validierung	80.2%	2.5%	78.0%	1.8%	0.33	0.07
Test	59.2%	4.7%	23.5%	4.3%	1.29	0.25

C) Ergebnisse der selbstüberwacht trainierten Encoder

C.1) 0.1-45Hz-Bandpassfilter

Augmentationsfaktor	Validierung		Test					
	aug-TA		aug-TA		knn-A		knn-MAE	
	Mittel	SD	Mittel	SD	Mittel	SD	Mittel	SD
1	96.8%	1.0%	96.4%	0.1%	22.9%	0.1%	1.28	0.02
2	94.8%	1.5%	92.8%	0.0%	24.1%	0.6%	1.26	0.01
3	87.9%	1.7%	83.2%	0.8%	24.1%	0.6%	1.28	0.02

C.2) 0.1-55Hz-Bandpassfilter

Augmentationsfaktor	Validierung		Test					
	aug-TA		aug-TA		knn-A		knn-MAE	
	Mittel	SD	Mittel	SD	Mittel	SD	Mittel	SD
1	95.7%	0.3%	81.4%	0.3%	24.8%	1.4%	1.25	0.01
2	86.4%	0.5%	67.6%	1.2%	25.2%	0.6%	1.22	0.01
3	80.8%	2.5%	59.9%	3.3%	24.9%	0.8%	1.23	0.01

D) Ergebnisse der Klassifikationsnetze

D.1) 0.1-45Hz-Bandpassfilter

	knn-A		head-A		knn-MAE		head-MAE	
	Mittel	SD	Mittel	SD	Mittel	SD	Mittel	SD
Überwacht	30.3%	5.6%	31.4%	8.3%	1.16	0.21	1.17	0.25
Selbstüberwacht	23.7%	0.7%	24.9%	1.5%	1.27	0.02	1.26	0.04

D.2) 0.1-55Hz-Bandpassfilter

	knn-A		head-A		knn-MAE		head-MAE	
	Mittel	SD	Mittel	SD	Mittel	SD	Mittel	SD
Überwacht	25.1%	0.3%	25.0%	0.0%	1.12	0.25	1.0	0.0
Selbstüberwacht	25.0%	0.7%	26.1%	2.8%	1.23	0.02	1.21	0.03