NeuroIDBench: An open-source benchmark framework for the standardization of methodology in brainwave-based authentication research

Avinash Kumar Chaurasia\textsuperscript{a}, Matin Fallahi\textsuperscript{b,\textdagger}, Thorsten Strufe \textsuperscript{b}, Philipp Terhörst \textsuperscript{a}, Patricia Arias Cabarcos \textsuperscript{a,\textdagger}

\textsuperscript{a} University of Paderborn, Paderborn, Germany
\textsuperscript{b} KASTEL Security Research Labs - KIT, Karlsruhe, Germany

\begin{abstract}
Biometric systems based on brain activity have been proposed as an alternative to passwords or to complement current authentication techniques. By leveraging the unique brainwave patterns of individuals, these systems offer the possibility of creating authentication solutions that are resistant to theft, hands-free, accessible, and potentially even revocable. However, despite the growing stream of research in this area, faster advance is hindered by reproducibility problems. Issues such as the lack of standard reporting schemes for performance results and system configuration, or the absence of common evaluation benchmarks, make comparability and proper assessment of different biometric solutions challenging. Further, barriers are erected to future work when, as so often, source code is not published open access. To bridge this gap, we introduce NeuroIDBench, a flexible open source tool to benchmark brainwave-based authentication models. It incorporates nine diverse datasets, implements a comprehensive set of pre-processing parameters and machine learning algorithms, enables testing under two common adversary models (known vs unknown attacker), and allows researchers to generate full performance reports and visualizations. We use NeuroIDBench to investigate the shallow classifiers and deep learning-based approaches proposed in the literature, and to test robustness across multiple sessions. We observe a 37.6\% reduction in Equal Error Rate (EER) for unknown attacker scenarios (typically not tested in the literature), and we highlight the importance of session variability to brainwave authentication. All in all, our results demonstrate the viability and relevance of NeuroIDBench in streamlining fair comparisons of algorithms, thereby furthering the advancement of brainwave-based authentication through robust methodological practices.
\end{abstract}

1. Introduction

In today’s rapidly changing security landscape, the authentication process is a crucial element of access control \cite{1}. The increasing impracticality and safety concerns associated with conventional authentication methods, such as passwords \cite{2,3}, are driving the development of novel biometric solutions. In this domain, user recognition based on brain activity has gained increasing attention \cite{4}, especially with advancements in consumer-grade EEG (Electroencephalogram) technology \cite{5–8}. A notable application scenario is Extended Reality (XR), where seamless and secure biometric authentication systems are urgently needed \cite{9}. Besides, brain biometrics bring broader benefits, such as enabling hands-free authentication, resistance to observation, inherent liveness detection, broad universality, and potential for revocability \cite{4,10,11}.

Unfortunately, the research in brainwave-based authentication is beset by problems of generalizability, comparability, and reproducibility. To begin, dataset sharing is sharply curtailed by ethical considerations and privacy laws. Therefore, many studies are based on self-collected undisclosed datasets, which often leads to results that are potentially overfitted or optimized to the specific characteristics of those datasets, with uncertain applicability to other, independent datasets. Also, the size of the datasets used in studies is severely limited by the complexity and resource intensiveness of broad data collection. In fact, most brainwave-based authentication studies have much fewer than 50 subjects \cite{4,12}, so that the evaluations are normally conducted...
on very small datasets. Consequently, the models and hyperparameters are prone to overfitting, which means the reported results will likely fail to generalize. To make matters worse, too many authors either neglect making available their source code or provide code which is inoperable, with the consequence that researchers are prevented from reproducing and building upon previous work. Clearly, under circumstances such as these, our subcommunity of authentication researchers will not make any significant progress. That is why the time has come for us to consolidate our practice and advance the research towards consistent, reproducible, and generalizable findings. To address this gap, we make the following contributions:

1. We present NeuroIDBench, the first comprehensive open-source benchmark tool designed to assist researchers in evaluating their brainwave-based authentication approaches. NeuroIDBench is engineered to allow for the easy and flexible integration of new datasets and methods, enabling fair comparability. NeuroIDBench incorporates nine public datasets, which together contain a substantial number of subjects (n = 285, Table 2) and multi-session recordings. To determine the impact of preprocessing on performance, NeuroIDBench allows to apply different sample rejection thresholds as well as sample length parameters (Section 4.1) and feature extraction methods (Section 4.2). Also, it implements a baseline with the most popular authentication algorithms in the state of the art, including shallow classifiers and deep learning-based approaches. Lastly, it incorporates two attack scenarios under which to test the algorithms [13]: (i) known attacker, where the attacker is known to the authentication model (i.e., enrolled in the system), and, (ii) unknown attacker, a more realistic situation in which the attacker biometric data has not been seen by the system previous to the attack.

2. We use NeuroIDBench to build a comprehensive benchmark of popular brainwave-based authentication algorithms, providing the first unbiased comparison of approaches. Namely, we compare six shallow classifiers: SVM (Support Vector Machine), Random Forest, KNN (K-Nearest Neighbors), LDA (Linear Discriminant Analysis), Naive Bayes (NB), and Logistic Regression (LR). Besides this benchmark, which covers the biggest share of solutions in the literature, we provide a second benchmark for the recent stream of work on deep learning-based approaches. Specifically, NeuroIDBench implements the representative similarity-based technique using Twin Neural Networks (Section 4.5). Beyond performance across an extensive list of publicly available EEG datasets, we focus our comparative analysis on two under-explored aspects in current research: How robust are different solutions in multi-session scenarios? (Section 4.4) and What is the impact of the adversary model? (Section 4.3)

Among the key findings in the analysis we observe that Random Forest consistently outperforms other classifiers and its performance is comparable to that of Twin Neural Networks. As the latter brings not only better performance but scalability advantages, we recommend more research on deep learning-based approaches to move forward, which is also the path followed in other biometric communities [14–16]. Furthermore, our results confirm that integrating Power Spectral Density (PSD) with an Autoregressive (AR) model of order 1 stands out as the most effective feature extraction approach. When investigating the impact of the adversary model, our results indicate that the average Equal Error Rate (EER) for known attackers is 2.87%, while for unknown attackers, it is 4.6%. This trend in performance degradation underscores the need to evaluate brainwave authentication solutions under unknown attacker scenarios to provide a realistic performance indicator, a practice that is uncommon in current work. Variations in the performance across datasets confirm that brainwave-based authentication algorithms validated on single datasets are not representative, highlighting the need for more robust validation in the community. Finally, the observed disparities in the results for single-session versus multi-session authentication reveal a substantial gap that requires future research in this area.

2. Background and related work

This section lays out the main terminology and background information on brainwave-based biometrics and reviews related work in the area, discussing reproducibility.

2.1. Using brainwaves for biometric authentication

Brainwaves, or EEG signals, represent the electrical activity produced by neuron interactions in the brain [17,18]. These signals are commonly recorded from the scalp using specialized sensors. Originally developed for medical purposes [19], such as diagnosing neurological disorders [20], EEG technology has expanded its applications significantly. Its usage now extends to Brain-Computer Interfaces (BCI), facilitating direct interaction between the human brain and external devices [21], and enjoys acceptance in the consumer market, especially for gaming and health-related applications.

Identity verification, commonly known as authentication, involves a one-to-one comparison to confirm if an individual presenting a previously registered biometric trait is indeed the same person. In essence, it addresses the question: “Are you who you claim to be?” This process is crucial to differentiate legitimate users from imposters. Biometric authentication comprises two primary phases: enrollment and verification [22]. During enrollment, users’ unique biometric characteristics are captured and stored as templates or used to train and store a classification model for that user. Each template/user model is linked to a unique user identifier, such as a username. In the verification phase, the system compares the current biometric sample with the stored template or runs it through the classification model, corresponding to the claimed identity.

The brainwave-based authentication process, depicted in Fig. 1, unfolds through several key stages:

- **Raw EEG Data Acquisition:** This initial step captures the brain’s electrical activity. An effective way to elicit signals useful for authentication is measuring brain reactions to controlled stimuli, such as audio or images. These time-locked reactions, called Event-Related Potentials (ERPs), have shown unique features to individuate people [24] and are distinguished by their high Signal-to-Noise Ratio (SNR) [25,10]. Moreover, the ability to modify the stimuli associated with ERPs enhances the system’s revocability. This feature is invaluable in mitigating risks associated with potential data compromise [11]. NeuroIDBench focuses in ERP-based datasets, given their superior benefits with regard to alternative brainwave elicitation paradigms.

1 https://github.com/Avichaursia/NeuroIDBench.git
• **Signal Pre-processing:** The acquired raw EEG data is then refined through pre-processing. This may include filtering out noise, rejecting artifacts, and downsampling the signal, aiming to isolate the purest form of the EEG data for authentication purposes [4]. NeuroIDBench embeds a pre-processing pipeline based on best-practices for handling ERPs as used in the community [4].

• **Feature Extraction:** The pre-processed EEG signals are further analyzed to extract key features. These features can be directly obtained from the ERP EEG signal, such as power spectral densities, or generated as representation vectors learned by a neural network. NeuroIDBench implements both kinds of approaches.

• **User Model/Template Database:** This Database stores the user model learned from the collected features in order to perform classification, or the template of features representing the user.

• **Decision Maker:** In the authentication phase, the system evaluates the incoming EEG sample, against the stored model/template, applying similarity thresholds to decide if it belongs to a legitimate user or to an impostor. NeuroIDBench incorporates the most common techniques for machine learning model-based approaches (SVM, KNN, LDA, NB, LR, RF), as well as the main approach for template-based authentication using deep neural networks (Twin Neural Networks).

2.2. Adversary model

In addressing the security of brainwave-based authentication systems, we consider the zero-effort attacker as the primary threat model. This approach aligns with the standard threat model used in most brainwave authentication studies. A zero-effort attack is characterized by an adversary who attempts to gain unauthorized access using their own biometric data, without employing any advanced techniques to mimic or falsify another individual’s biometric trait [13]. In this context, the key challenge in protecting against zero-effort attacks lies in the biometric system’s ability to accurately distinguish between the biometric data of an actual user and that of the intruder. The system must be adept at minimizing false matches, where the adversary’s biometric data is erroneously accepted as that of a legitimate user.

In our consideration of zero-effort attacks, we distinguish between two types of attackers: **known** and **unknown**. A known attacker refers to one whose biometric data has previously been exposed to the system, typically during the training phase. This exposure can influence the system’s learning and adaptation, potentially affecting its ability to accurately identify or reject these attackers later. On the other hand, an unknown attacker is one whose biometric data has never been introduced to the system. This lack of prior exposure means that the system has no learning or adaptation based on this attacker’s data, making it a more realistic and challenging scenario. We expect that unknown attackers represent a higher risk to the system, as they test the system’s ability to authenticate users based purely on learned patterns by an adversary who attempts to gain unauthorized access using their biometric data (30%) or source code (40%) even in the papers published at long-lived” subcommunity of machine learning security fail to provide data (30%) or source code (40%) even in the papers published at the top-tier security conferences. The authors found out that a full 80% of artifacts are incapable of reproducing the reported results. It is therefore unsurprising that similar problems should plague the new and emerging subcommunity in brainwave-based authentication. Nevertheless, these issues are intensified in brainwave authentication due to the relatively small size of each dataset, heightening the risk of overfitting.

Meanwhile, researchers in the field of Brain-Computer Interface (BCI) encounter similar challenges. Hence, Jayaram and Barachant [26] proposed a benchmark framework for investigating common BCI algorithms across 22 public EEG datasets with over 250 participants. This framework aims to address issues of reproducibility, lack of open-source resources, and result compatibility. However, their study did not cover authentication algorithms. Therefore, it is vital for brainwave authentication research to adopt an open-source approach and to extensively employ existing public datasets. It is crucial to take such steps to enhance the robustness of research in this field and to promote a deeper, more comprehensive exploration of brainwave authentication techniques.

3. NeuroIDBench

As illustrated in Fig. 2, NeuroIDBench is organized into five integral components: dataset, preprocessing, feature extraction, evaluation, and analysis. Our benchmarking workflow is influenced by MOABB (Mother of all BCI benchmarks) [26] work. In line with our research...
Aims, we have tailored and refined their methodology to create a customized benchmarking suite specifically designed for evaluating and analyzing brainwave authentication systems. In the following, we detail NeuroIDBench’s components, justifying the design choices.

### 3.1. Dataset loading

In response to the need for flexible dataset management in EEG research, we have developed an interface aimed at streamlining the handling of diverse datasets with various formats, thereby reducing the burden on researchers. In the past, this field has been constrained by the laborious task of manually importing and formatting each dataset individually, often limiting researchers to using only one or two datasets in their analyses due to these logistical challenges. To overcome this, our solution is a straightforward Python component that automatically downloads and processes nine distinct EEG datasets using the MNE Python package, a recognized and open-source brainwave library. This tool not only simplifies the integration of these existing datasets but also allows for the easy adoption of new datasets into the benchmark by following established examples and specifying the relevant event IDs. The expected outcome of this innovation is a significant enhancement in the use of diverse datasets in research, promoting the avoidance of overfitting to specific datasets and fostering more generalizable and robust scientific findings. This development represents a leap forward in EEG-based research, offering researchers a more efficient and comprehensive approach to dataset management and analysis.

In choosing our dataset, we focused on Event-Related Potential (ERP) paradigms, particularly the P300 and N400, which are the most

---

### Table 2

<table>
<thead>
<tr>
<th>Paradigm</th>
<th>Dataset</th>
<th>Year</th>
<th>Subjects</th>
<th>EEG Device</th>
<th>Channels</th>
<th>S.Rate</th>
<th>Sessions</th>
</tr>
</thead>
<tbody>
<tr>
<td>N400</td>
<td>Pijnacker et al. [39]</td>
<td>2017</td>
<td>45</td>
<td>actiCap</td>
<td>32</td>
<td>500 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>Draschkow et al. [40]</td>
<td>2018</td>
<td>40</td>
<td>BrainAmp, actiChamp</td>
<td>64</td>
<td>1000 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>Marzecová et al. [41]</td>
<td>2018</td>
<td>18</td>
<td>BrainAmp</td>
<td>59</td>
<td>500 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>Mantegna et al. [42]</td>
<td>2019</td>
<td>31</td>
<td>BrainAmp, EasyCap</td>
<td>65</td>
<td>500 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>ERPCORE: N400 [36]</td>
<td>2021</td>
<td>40</td>
<td>Biosemi</td>
<td>30</td>
<td>1024 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>Hodapp and Rabovky [43]</td>
<td>2021</td>
<td>33</td>
<td>BrainAmp</td>
<td>64</td>
<td>1000 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>Rabs et al. [44]</td>
<td>2022</td>
<td>38</td>
<td>BrainVision</td>
<td>26</td>
<td>500 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>Schoknecht et al. [45]</td>
<td>2022</td>
<td>38</td>
<td>ActiCap, ActiChamp</td>
<td>58</td>
<td>500 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>Tofolfo et al. [46]</td>
<td>2022</td>
<td>24</td>
<td>Biosemi</td>
<td>128</td>
<td>512 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>Lindborg et al. [47]</td>
<td>2022</td>
<td>40</td>
<td>BrainVision</td>
<td>64</td>
<td>2046 Hz</td>
<td>1</td>
</tr>
<tr>
<td>N400</td>
<td>COGBCI: Flanker [48]</td>
<td>2023</td>
<td>29</td>
<td>ActiCap, actiChamp</td>
<td>64</td>
<td>512 Hz</td>
<td>3</td>
</tr>
<tr>
<td>N400</td>
<td>Stone et al. [49]</td>
<td>2023</td>
<td>64</td>
<td>TMSi Refa</td>
<td>32</td>
<td>512 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>BrainInvaders12 [50]</td>
<td>2012</td>
<td>25</td>
<td>NeXus-32</td>
<td>16</td>
<td>128 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>BrainInvaders13a [51]</td>
<td>2013</td>
<td>24</td>
<td>g.GAMMAcap</td>
<td>16</td>
<td>512 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>BrainInvaders14a [52]</td>
<td>2014</td>
<td>64</td>
<td>g.Sahara</td>
<td>16</td>
<td>512 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>BrainInvaders14b [53]</td>
<td>2014</td>
<td>37</td>
<td>g.GAMMAcap</td>
<td>32</td>
<td>512 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>Gao et al. [54]</td>
<td>2014</td>
<td>30</td>
<td>Neuroscan</td>
<td>12</td>
<td>500 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>BrainInvaders15a [55]</td>
<td>2015</td>
<td>50</td>
<td>g.GAMMAcap</td>
<td>32</td>
<td>512 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>BrainInvaders15b [56]</td>
<td>2015</td>
<td>44</td>
<td>g.GAMMAcap</td>
<td>32</td>
<td>512 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>Moulsek et al. [57]</td>
<td>2017</td>
<td>250</td>
<td>BrainVision</td>
<td>3</td>
<td>n.a.</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>Hubner et al. [58]</td>
<td>2017</td>
<td>13</td>
<td>BrainAmp DC</td>
<td>31</td>
<td>1000 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>Sosulski and Tangermann [59]</td>
<td>2019</td>
<td>13</td>
<td>BrainAmp, EasyCap</td>
<td>31</td>
<td>1000 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>Lee et al. [60]</td>
<td>2019</td>
<td>54</td>
<td>BrainAmp</td>
<td>62</td>
<td>1000 Hz</td>
<td>2</td>
</tr>
<tr>
<td>P300</td>
<td>Simões et al. [61]</td>
<td>2020</td>
<td>15</td>
<td>g.tec</td>
<td>8</td>
<td>250 Hz</td>
<td>7</td>
</tr>
<tr>
<td>P300</td>
<td>Goncharenko et al. [62]</td>
<td>2020</td>
<td>60</td>
<td>NVX-52</td>
<td>8</td>
<td>500 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>Chatroudi et al. [63]</td>
<td>2021</td>
<td>24</td>
<td>g.tec</td>
<td>64</td>
<td>1200 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>Cattan et al. [64]</td>
<td>2021</td>
<td>21</td>
<td>g.USBallop, g.tec</td>
<td>16</td>
<td>512 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>ERPCORE: P300 [36]</td>
<td>2021</td>
<td>40</td>
<td>Biosemi</td>
<td>30</td>
<td>1024 Hz</td>
<td>1</td>
</tr>
<tr>
<td>P300</td>
<td>Won et al. [65]</td>
<td>2022</td>
<td>55</td>
<td>Biosemi</td>
<td>32</td>
<td>512 Hz</td>
<td>1</td>
</tr>
</tbody>
</table>
well-known and easier to elicit. The P300 reaction is triggered with the ‘oddball’ paradigm, i.e., including a rare stimulus in a sequence [66], while the N400 is linked to semantic processing [67]. Our selection methodology involved a thorough review of publicly accessible datasets utilizing these ERP paradigms. We carefully examined and assessed over 29 datasets (see Table 2). Ultimately, we chose 9 datasets for further investigation. This selection included two multi-session datasets to examine performance over multiple sessions [48,60], five datasets with over 30 subjects to evaluate performance in datasets with a relatively large number of samples [42,36,55,65], and two datasets with just 13 subjects to explore performance in smaller datasets [58,59].

Our prioritization focused on datasets providing raw (unprocessed) data with ample event information, facilitating the extraction of relevant P300 or N400 samples. This emphasis led to the implementation of a condition favoring datasets with raw data, allowing us to uniformly apply standardized pre-processing, feature extraction, and authentication steps across all selected datasets. This uniform procedure is crucial for evaluating their performance under comparable experimental conditions, an unattainable task without access to unprocessed raw data. Consequently, datasets offering solely pre-processed data were excluded from our study. Additionally, the necessity for comprehensive event information was paramount; without it, the assurance of working with the correct events, intricately associated with the elicitation of P300 and N400, would have been compromised.

3.2. Pre-processing

In this module, we conduct pre-processing on the standardized MNE Raw data. Various methods exist for cleansing artifacts; however, the procedures must remain consistent to ensure the validity of comparisons between algorithms or datasets [26]. We adhere to established best practices commonly employed in pre-processing methodologies within brainwave authentication studies [4]. To cleanse EEG artifacts, we begin by implementing bandpass filtering in the range of 1 to 50 Hz. Within brainwave authentication studies [4], we recognize the necessity for in-brainwave datasets providing raw (unprocessed) data with ample event information, facilitating the extraction of relevant P300 or N400 samples. This emphasis led to the implementation of a condition favoring datasets with raw data, allowing us to uniformly apply standardized pre-processing, feature extraction, and authentication steps across all selected datasets. This uniform procedure is crucial for evaluating their performance under comparable experimental conditions, an unattainable task without access to unprocessed raw data. Consequently, datasets offering solely pre-processed data were excluded from our study. Additionally, the necessity for comprehensive event information was paramount; without it, the assurance of working with the correct events, intricately associated with the elicitation of P300 and N400, would have been compromised.

3.3. Feature extraction

This module is designed to extract features from pre-processed EEG data. It specifically focuses on identifying characteristics in both the temporal and frequency domains. More precisely, it uses Autoregressive (AR) coefficients to represent time-domain properties and calculates Power Spectral Density (PSD) to capture frequency domain features. These calculated features, namely AR coefficients and PSD, are subsequently harnessed as inputs for facilitating authentication through shallow classifiers. In contrast, for similarity-based authentication, the module employs the Twin Neural Network (TNN) to generate feature embeddings. These embeddings, produced by the TNN, play a pivotal role as inputs for similarity-based authentication approaches.

To compute AR coefficients, our system fits AR models to the pre-processed samples, employing a one-second duration of time series data [23]. The estimation of these AR coefficients utilized the Yule–Walker method [68]. Determining the ideal order for AR modeling poses a challenge, as higher orders escalate computational demands, while exceedingly low orders inadequately represent the signal [69]. To address this, our system allows users the flexibility to define their preferred AR order, enabling a tailored approach aligned with their specific requirements.

The PSD of each sample is computed across different frequency bands, namely low (1–10 Hz), α (10–13 Hz), β (13–30 Hz), and γ (30–50 Hz), utilizing the Welch’s periodogram algorithm [23]. The calculation involved first determining the PSD for each frequency point in the 1-second sample. For our PSD computation, we used four time windows that were all the same size. These windows were applied to a 1-second ERP sample, with a 50% overlap between each window. The inclusion of the time window factor was crucial to segregate the authentic frequency modulation of the EEG induced by attention from any artifacts that the attentional modulation of ERPs might have induced [70]. Subsequently, we computed the average PSD within the specified frequency ranges, enabling us to determine the average power spectrum of the low, α, β, and γ frequency bands.

Furthermore, NeuroidBench implements a deep learning based approach using a TNN with triplet loss function to transform time-series EEG data into compact brain embeddings. Building on the work of Fallahi et al. [27], our TNN structure integrates three Convolutional Neural Network (CNN) branches, each featuring five convolutional layers, as illustrated in Fig. 3. These branches received 1-second samples as input, structured in a two-dimensional array where rows correspond to channel indices, and columns represent discrete-time measurements. After each convolutional layer, we applied an average pooling layer to reduce the input vectors’ dimensionality while preserving the unique characteristics of each brainwave.

However, acknowledging the diverse nature of EEG datasets and the variability in research requirements, we have extended the flexibility of our framework; To accommodate varying preferences and dataset characteristics, users can conveniently adjust critical training parameters such as batch size, the number of epochs for training, and the learning rate within the tool. Also, the framework allows users to integrate...
their neural network architectures. This flexibility accommodates diverse methodologies for brainwave authentication studies, empowering researchers to assess and compare various deep-learning approaches across a wide range of EEG datasets within our tool.

3.4. Evaluation

While most EEG studies assess brainwaves in a single-session scenario [71,72,23,29,34,35], it is crucial to examine EEG robustness over time. A realistic and often overlooked approach is the multi-session scenario. To highlight this gap, we provide both single-session and multi-session evaluation scenarios in our benchmark tool as follows.

**Single-Session Evaluation:** Under the single-session evaluation, the training and testing of the features are done utilizing the recorded data from a single session. To avoid overfitting and increase the reliability of our authentication system, cross-validation schemes have been employed in both known and unknown attacker scenarios.

For the known attacker scenario, we implemented a Stratified cross-validation approach with $k = 4$ to divide the single-session data into training and testing sets. This choice ensures the representation of features from both classes in both training and testing data across each fold. We take the step of eliminating users with fewer than four samples from the datasets to guarantee sufficient samples for both training and testing [23]. Conversely, the unknown attacker approach adopts the GroupKFold cross-validation strategy with $k = 4$, where grouping is based on SubjectID, resulting in non-overlapping training and testing sets for users in each cross-validation round. Evaluation metric results are aggregated across all folds, providing averaged and comprehensive reports. In datasets like COGBCI and Lee2019, which had multiple sessions, we evaluate each session separately, subsequently averaging the results from all the sessions as single-session evaluation results. Multi-session evaluation, described later in this section.

Additionally, we actively apply z-scaler normalization to both training and testing data to address potential issues arising from variations in feature scales, ensuring equitable contribution from all features during model training [72]. This approach aligns with established best practices in machine learning [73], reinforcing the reliability and generalization capability of our EEG-based authentication system.

**Multi-Session Evaluation:** The multi-session evaluation methodology was applied to two EEG datasets, namely COGBCI [48] and Lee2019 [60]. COGBCI consists of three sessions separated by day intervals, while Lee2019 includes two sessions separated by week intervals. A cross-validation strategy was employed to address overfitting. This involved dividing the attacker subjects for the shallow classifiers and multi-session evaluation scenarios in our benchmark tools as follows.

**Multi-Session Evaluation:** The multi-session evaluation methodology was applied to two EEG datasets, namely COGBCI [48] and Lee2019 [60]. COGBCI consists of three sessions separated by day intervals, while Lee2019 includes two sessions separated by week intervals. A cross-validation strategy was employed to address overfitting. This involved dividing the attacker subjects for the shallow classifiers and multi-session evaluation scenarios in our benchmark tools as follows.

**Fig. 4.** Impact of varying sample duration, ranging from 1 to 2 seconds, on the efficacy of the seven authentication algorithms. These algorithms are applied across nine datasets, with the evaluation metric employed being the EER. The evaluation was conducted under the unknown attacker scenario.

Proper metrics are essential for comparing different methods from the viewpoint of uniqueness. We have chosen to report EER (Equal Error Rate) and FNMR (False Non-Match Rate) at various FMR (False Match Rate) levels, as recommended by the biometric authentication community [74,75] and international standards [76,77]. The FMR indicates the likelihood of an unauthorized user being mistakenly authenticated by the system, reflecting the system's vulnerability to a zero-effort attack. The goal is to achieve a low FMR while maintaining an acceptable FNMR. A high FNMR can result in legitimate users being repeatedly denied access, negatively impacting the device's usability. Therefore, it is crucial to balance FMR and FNMR to ensure robust security without compromising user experience. The EER represents the point, threshold, where the FMR and FNMR are equal.

Furthermore, this benchmarking framework is developed with a primary focus on ensuring ease of use. Our goal is to enable users to efficiently utilize the framework, even without a thorough grasp of the
complex technical aspects of the Python programming language. To accomplish this, we have created a user-friendly benchmarking script that proficiently examines a configuration file written in a straightforward YAML interface. This configuration file serves as a control panel, enabling users to specify a wide range of parameters and settings. This system effortlessly automates the complex processes of data extraction, pre-processing, feature extraction, and classification, as illustrated in Fig. 2. This streamlined approach eliminates the need for users to delve into intricate programming complexities. In Appendix B, the procedural steps for tool installation are detailed, accompanied by illustrative examples of configuration files for the automated creation of brainwave authentication pipelines.

4. Benchmarking results and discussion

We began with preprocessing and feature extraction to identify the best parameters for subsequent experiments. Subsequently, we compared known and unknown attacker models to ascertain the increased difficulty associated with unknown problems. This was followed by an investigation into single and multi-session performances with unknown attacker scenarios. Additionally, we examined the effectiveness of two primary approaches: shallow classifiers and similarity approaches.

4.1. Preprocessing

This study examines two critical aspects of electroencephalography (EEG) preprocessing: sample duration and sample rejection. Typically, researchers utilize a one-second sample, spanning from 0.2 seconds before to 0.8 seconds after a stimulus. We explore various sample durations, specifically [1.0, 1.2, 1.4, 1.6, 1.8, 2.0] seconds. It is hypothesized that longer durations may enhance results due to the inclusion of more information. However, considering the primary objective of event-related potentials (ERP) is to introduce stimuli for signal-to-noise ratio (SNR) enhancement, extended samples might inadvertently incorporate noise data. Another preprocessing step investigated is sample rejection, based on peak-to-peak (PTP) signal amplitude. We establish a rationale behind PTP sample rejection as the assumption that samples with high PTP values are likely to be noisy and their removal might enhance the performance of brainwave-based authentication systems. Our investigation focuses on determining whether this preprocessing step can indeed improve the efficacy of brainwave authentication and, if so, identifying the optimal threshold level for PTP sample rejection.

Fig. 5 displays EER across various PTP rejection thresholds. Results show that for classifiers like LR, LDA, and NB, a lower threshold results in a reduced EER. In contrast, RF and SVM displayed no specific trend, and their performance without sample rejection already surpassed other shallow classifiers. For the Twin Neural Network, sample rejection had either no effect or a negative impact. Overall, the study found no significant improvement attributable to PTP rejection.

4.2. Feature extraction

We explored Power Spectral Density (PSD) and Autoregressive (AR) models of different orders as feature extraction methods typically employed in shallow classifiers. Fig. 6 illustrates the EER for various configurations of feature extraction across different classifiers. The findings suggest that the combination of PSD with AR of order 1 yields superior performance compared to other combinations. Following this, PSD features alone demonstrate promising results, whereas AR on its own fails to show stable and robust outcomes. Interestingly, in the BrainInVaders15a dataset, the AR of order 1 outperformed most classifiers. In the ERPCORE_P300 dataset, PSD was the predominant feature leading to superior performance across most classifiers. Additionally, in the COG_BCI dataset, a higher AR order demonstrated better performance compared to lower AR orders. These results substantiate our assertion that analyzing a single dataset can yield results that are not generalizable to other datasets. However, it is noteworthy that across all these datasets, the combination of PSD with AR order 1 consistently appears among the top-performing features. Consequently, we will utilize these features in the subsequent sections of this paper as the default feature set for shallow classifiers.
4.3. Known and unknown attacker approach

The known attacker model represents an unrealistic scenario; however, given its existence in some previous studies, we investigated the performance gap between known and unknown attacker scenarios. Fig. 7 displays the EER for these two scenarios, based on a sample duration of 1 seconds to ensure comparability with previous work. This analysis was conducted without sample rejection and utilized PSD combined with AR order 1 as the feature extraction method for shallow classifiers.

The results reveal that the mean EER across datasets degraded by 58.44% for KNN, 275.60% for LDA, 383.91% for LR, 5.83% for NB, 75.94% for RF, 66.61% for SVM, and 75.88% for the Twin neural network approach. Notably, in some cases, the EER increased several-fold. These findings suggest that results derived from the known attacker model can be misleading, emphasizing the need for researchers.
to exercise caution and potentially avoid using this model in their analyses.

4.4. Single session vs. Multi session authentication

For the practical application of brainwave authentication, it is essential to develop a model capable of handling multi-session authentication. However, there is a notable limitation in the availability of datasets for multi-session studies. Our investigation focuses on comparing single-session and multi-session authentication to understand the performance gap between these two approaches. We expected an increase in EER for multi-session authentication, as it represents a more complex challenge. This increase in difficulty is attributed to additional noise factors inherent in multi-session settings. Additionally, variations in EEG electrode placement across sessions and possible changes in brain states add to the complexity, making multi-session authentication a more challenging task compared to single-session scenarios.

Fig. 8 presents the results for multi-session authentication, where the results indicate a significant increase in EER compared to the single-session results shown in Fig. 7. Notably, LDA and LR, which were not among the best performers in the single-session scenario, show more promising results in the multi-session context. These findings highlight the need for further research focused on multi-session scenarios to better understand and improve authentication performance in these more complex settings. The average EERs of 21% and 30% in multi-session scenarios indicate limitations in the practical implementation of a brainwave biometric authentication system, especially considering the False Rejection Rate (FNMR) at False Acceptance Rates (FMR) of 1%, 0.1%, and 0.01%. These rates were respectively 72.61%, 86.70%, 89.50% for the Lee2019 dataset and 84.43%, 94.05%, 94.09% for the COG_BCI_Flanker dataset (Table 3), suggesting the need for improved authentication performance in these more complex scenarios.

Fig. 8 presents the results for multi-session authentication, where the results indicate a significant increase in EER compared to the single-session results shown in Fig. 7. Notably, LDA and LR, which were not among the best performers in the single-session scenario, show more promising results in the multi-session context. These findings highlight the need for further research focused on multi-session scenarios to better understand and improve authentication performance in these more complex settings. The average EERs of 21% and 30% in multi-session scenarios indicate limitations in the practical implementation of a brainwave biometric authentication system, especially considering the False Rejection Rate (FNMR) at False Acceptance Rates (FMR) of 1%, 0.1%, and 0.01%. These rates were respectively 72.61%, 86.70%, 89.50% for the Lee2019 dataset and 84.43%, 94.05%, 94.09% for the COG_BCI_Flanker dataset (Table 3), suggesting the need for improved authentication performance in these more complex scenarios.

The comparison of our results will initially focus on the results published by Arias et al. [24] and Fallahi et al. [27]. This is due to the establishment of our benchmark, which is based on their implementations. Moreover, both studies utilized the ERPCORE [36] P300 and N400 datasets, similar to our approach. Arias et al. [24] investigate the performance of different shallow classifiers within single-session brainwave authentication. The results indicate 1.9% EER for N400 and 3% EER for P300 paradigm with a sample rejection of 120 μV and RF classifier, we observe the same results where we achieved 2.9% EER for P300 and 1.13% EER to N400 paradigm with the sample rejection of 100 μV and RF classifier. Also, Fallahi et al. [27] acquired 2.01% and 1.37% respectively for P300 and N400 ERPCORE and 0.14 for P300:bi2015a dataset [55], where we obtained %1.53, 0.83%, and 0.43 respectively. As can be observed the results are fairly similar and trends are the same. The small difference in results could be from a different random seed, version of libraries, and some evaluation parameters like epoch rejection rate for shallow classifiers and number of epochs for twin neural networks. To enhance the reproducibility of our result, we provided a docker container in the GitHub of NeuroIDBench.

Preprocessing: Our results about the impact of sample duration corroborate prior research on shallow classifiers, which indicates that longer sample durations can result in reduced EER or increased accuracy [24,78–80]. However, the current version of the twin neural network requires modification to effectively utilize longer samples.

Feature Extraction: The results of our feature extraction analysis demonstrate that PSD features notably outperform AR features, corroborating findings from Huang et al. [81]. However, there is a lack of consensus in the literature regarding the optimal order for AR features. Studies by Arias et al. [23] and Brigham [82] suggest that lower AR orders yield better results, whereas research by Kaewwits et al. [83] and Zhang et al. [69] indicates improved outcomes with higher orders. Our findings align with the former, showing that lower-order AR features are more effective in 8 out of 9 datasets analyzed.

Known and Unknown attacker: In exploring scenarios involving known and unknown attackers, Wu et al. [84] conducted research but found no notable performance decline in the unknown attacker scenario, and even reported an improvement. Similarly, Panzino et al. [85] studied the same issue and observed only a 5% decrease in accuracy. Contrarily, Arias et al. [23] and Fallahi et al. [27] reported an increase in Equal Error Rate (EER) ranging from 10% to 17.5 times. Our results indicate that the unknown attacker scenario is consistently more effective than the known attacker scenario, although the extent of this difference varies across different datasets and models. An average calculation across nine datasets and seven models shows that the EER for unknown attackers is 60.2% higher than known attacker scenarios.

Single and Multi-session: Multi-session analysis is crucial for the practical application of brainwave authentication, yet it presents heightened challenges compared to single-session scenarios. In multi-session environments, the availability of public datasets is notably
Scarce, and the practice of open-source sharing becomes more constrained. Results from various studies, such as Maiorana [30], demonstrate reasonable EER as low as 4.8%. Seha et al. [86] also reported a marginal difference of less than 0.5% EER between single and multi-session scenarios under specific feature extraction methods and LDA classifiers. Furthermore, Wu et al. [84] observed a 1.3% improvement in false rejection rate performance in the second session compared to the first session in single-session scenarios. In contrast, our research reveals a significant disparity between single-session and multi-session results, with our best outcome showing an 11.99% EER. This finding aligns with Huang et al. [81], who also reported a substantial difference between within-session and cross-session evaluations. However, it is important to note that the studies by Maiorana [30], Seha et al. [86], and Wu et al. [84] do not provide public datasets or open-source code for other researchers to utilize and build upon. Therefore, there is an urgent need for more open-source and publicly available datasets in multi-session studies in this field. In fact, understanding the exact state of this challenge is crucial to determine what steps are necessary to advance the field.

### 5. Limitations

The two main limitations observed while conducting research to develop our benchmark are the shortage of publicly available datasets and the evolving ethical and privacy concerns in the deployment of EEG-based biometric systems. These challenges underscore critical areas for future research.
future research and development to enhance the reliability and ethical assurance of EEG authentication.

Publicly Available Datasets: Currently available public datasets are not designed for authentication purposes. They often have a low number of subjects, usually fewer than 70, and mostly provide data from a single session, making it difficult to evaluate the robustness of EEG authentication over time (Table 2). Additionally, most datasets use medical-grade devices for data collection, complicating performance estimation in real-world scenarios. Therefore, there is a need to collect authentication datasets that include a larger number of subjects (over 500), with at least three sessions, and various environmental conditions, such as variations in noise, lighting, hair length, and emotional states. This will enable proper training and evaluation of models and allow for comparisons with well-known biometrics such as face recognition models.

Ethical and Privacy Implications: As EEG authentication research advances towards practical applications, increasing attention is being paid to the ethical aspects of collecting and using brain data for authentication. Previous studies have looked into privacy concerns of researchers [88,89] and users [90,91], identifying the need for protecting these data. The main issue is that EEG can reveal highly sensitive personal information, including emotional states [92], medical conditions [93], attention levels [94], and gender [95]. Therefore, collecting brain data raises the risk of harmful data breaches and introduces new possibilities for abuse or misuse. For example, an honest-but-curious authentication provider could perform unauthorized behavioral monitoring or exploitation of brain reactions for commercial purposes. This potential threat is aggravated by the lack of user awareness regarding the sensitivity of brain data when using commercial BCIs [96]. Beyond privacy-related concerns, there is a risk of bias in deployed EEG-based authentication systems if not trained with a diverse set of people. This type of issue has been observed in face recognition software that performs poorly when used by subjects who belong to underrepresented groups [97].

To address these concerns, we recommend the following general guidelines for ethical practice. In terms of data collection for research, researchers must adhere to the ethical standards established in the Menlo Report [98] and inform subjects about potential risks through consent forms. For real-world EEG authentication, it is crucial to develop robust template protection methods to prevent authentication providers from accessing raw EEG data, protecting against potential leaks and inferences. Systems should be trained to ensure fairness and reduce algorithmic bias, which entails the collection of inclusive datasets that capture the diversity of all potential users. Similarly, broad studies on the social acceptance and diverse user needs should be conducted to guide a responsible human-centered development of EEG biometrics. Finally, it is important to implement effective transparency mechanisms beyond unusable privacy policies to communicate to BCI users what the collected data and privacy risks are.

6. Conclusion and future work

In conclusion, NeuroIDBench is presented as a comprehensive benchmarking tool for EEG authentication, particularly supporting ERP-based methods. As an open-source platform, it currently supports 9 datasets and 7 classifiers. It can easily be adapted to include new datasets and classifiers, allowing for the investigation of different methodological research questions. Our results highlight that epoch rejection does not significantly affect outcomes. PSD combined with AR order 1 emerges as a recommended default feature set for shallow classifiers. Known attacker evaluation can misleadingly indicate lower EER. Multi-session evaluation is significantly more challenging than single-session evaluation. Moreover, it appears that due to the limited availability of data, deep learning approaches do not consistently outperform shallow classifiers in brainwave-based authentication.

We encourage researchers to contribute to this open-source project, aiding in the development of more practical and diverse EEG-based authentication systems.
Fig. A.9. ROC for dataset BrainInvaders15a in Single Session Evaluation under unknown attacker Scenario.

Fig. A.10. ROC for dataset COG_BCI_Flanker in Single Session Evaluation under unknown attacker Scenario.

Fig. A.11. ROC for dataset ERPCORE_N400 in Single Session Evaluation under unknown attacker Scenario.

Fig. A.12. ROC for dataset ERPCORE_P300 in Single Session Evaluation under unknown attacker Scenario.

B.1. Establishing the running environment

The first step after extracting the tool from Github is to set up the running environment first. An optimal approach that involves creating a Python virtual environment dedicated to the framework, achieved through the creation of a Conda environment. This environment encapsulates all Python dependencies essential for executing the tool seamlessly. Alternatively, users may opt to containerize the Python package using Docker, thereby providing an isolated environment that encapsulates all dependencies required to execute the tool within the container. This approach ensures reproducibility and facilitates the seamless deployment of the benchmarking tool.

Setting up the Conda Environment

To successfully run this Python project, creating and configuring a suitable Python environment is essential. The following steps need to be followed to set up the environment:

1. **Python Installation**: Initially, verifying the presence of Python in our operating system is crucial. If Python is not pre-installed, it can be acquired straight from the official Python website. Subsequently, we adhere to the installation instructions outlined in the website, customized to suit our operating system. If Anaconda has been successfully installed, it is noteworthy to mention that the Anaconda installation often includes the Python programming language. If the action above is taken, it is possible to go to the subsequent stage.

2. **Anaconda Installation (if needed)**: Suppose Anaconda is not currently installed, and it is desired to utilize it to manage Python environments. In that case, it is possible to get the software by downloading it from the official website, which can be accessed at Anaconda website. Installation may be accomplished according to the instructions tailored to each operating system. Anaconda provides a user-friendly method for creating and administrating virtual environments through the use of

---

5 https://www.python.org/downloads/

6 https://www.anaconda.com/products/individual
Conda. This specific characteristic has significant value in data science and scientific computing initiatives.

3. Virtual Environment Creation: It is advisable to establish a virtual environment to segregate the dependencies of this specific project from other Python packages installed on our system. Virtual environments help maintain clean and distinct Python environments for individual projects. To create a virtual environment, follow these steps:

- **Navigate to Project Directory:** To begin, we access the terminal or command prompt and proceed to the project’s root directory by utilizing the `cd` command. For example: `cd /path/to/project`

- **Environment Configuration File:** Check if the project includes an environment configuration file. This file is typically named `environment.yml` or `requirements.txt` and lists the required Python packages and their versions.

- **Create Virtual Environment:** Subsequently, the requisite command is executed to generate a virtual environment by using the configuration file. An example of a command that may be used for Conda environments is the use of a `environment.yml` file:
  
  For example: `conda env create -f environment.yml`

  We can also utilize `requirements.txt` to create the virtual environment by using the pip command:

  ```shell`
  python -m venv venv
  source venv/bin/activate
  pip install -r requirements.txt
  ```

4. Activate the Virtual Environment: After the virtual environment has been established, proceed to activate it. Activation is a crucial process that guarantees using an isolated environment and its corresponding dependencies in our project. To activate the conda environment, it is necessary to utilize the proper command according to the operating system in use:

  For example: `conda activate master_thesis` (for MacOS/Linux)

**Setting up Docker Container**

To containerize a project, we typically use a tool like Docker. Here’s a step-by-step guide on how to do it:
1. Install Docker on the local machine if we have not already.
   Install docker from the official Docker’s website.

2. Create a dockerfile in the root directory of the framework.
   This file will contain the instructions for Docker to build our container.

3. Get the Python docker image python:3.8-slim-buster from Docker Hub.

4. Edit the dockerfile with the following steps:
   FROM python:3.8-slim-buster
   WORKDIR /app
   ADD ./app
   COPY requirements.txt ./
   RUN pip install –upgrade pip
   RUN apt-get update && apt-get install -y
   \liblapack-dev
   \libhdf5-dev
   \cython
   \pkg-config
   RUN pip install –no-binary=h5py h5py
   RUN pip install -r new_requirements.txt
   WORKDIR /app/neuroIDBench
   CMD ['python', 'run.py']

5. Build the docker image from the dockerfile, using the following command:
   docker build -t dockerfile.

B.2. YAML configurations

Once the Python environment or Docker container is developed, we need to edit the YAML configuration files that can be found as single_dataset.yml under the configuration_files folder. Following are some of the examples of the YAML configuration files that can be utilized to perform automated benchmarking and reproduce the results for the single dataset.

Listing 1: Benchmarking pipeline using the dataset’s default parameters and auto-regressive features with SVM classification

name: "BrainInvaders2015a"
dataset:
  - name: BrainInvaders2015a
    from: neuroIDBench.datasets
pipelines:
  "AR+SVM":
    - name: AutoRegressive
      from: neuroIDBench.featureExtraction
    - name: SVC
      from: sklearn.svm
      parameters:
        kernel: 'rbf'
        class_weight: "balanced"
        probability: True
Listing 2: Benchmarking pipeline using dataset's parameters and AutoRegressive order with SVM classification

name: "BrainInvaders2015a"

dataset:
- name: BrainInvaders2015a
  from: neuroIDBench.datasets
  parameters:
  - subjects: 10
  - interval: [-0.1, 0.9]
  - rejection_threshold: 200

pipelines:
"AR+SVM":
- name: AutoRegressive
  from: neuroIDBench.featureExtraction
  parameters:
  - order: 5
- name: SVC
  from: sklearn.svm
  parameters:
  - kernel: 'rbf'
  - class_weight: "balanced"
  - probability: True

Listing 3: Benchmarking pipeline for dataset BrainInvaders15a with AR and PSD features with classifier SVM

name: "BrainInvaders2015a"

dataset:
- name: BrainInvaders2015a
  from: neuroIDBench.datasets
  parameters:
  - subjects: 10
  - interval: [-0.1, 0.9]
  - rejection_threshold: 200

pipelines:
"AR+PSD+SVM":
- name: AutoRegressive
  from: neuroIDBench.featureExtraction
  parameters:
  - order: 5
- name: PowerSpectralDensity
  from: neuroIDBench.featureExtraction
- name: SVC
  from: sklearn.svm
  parameters:
  - kernel: 'rbf'
  - class_weight: "balanced"
  - probability: True

Listing 4: Benchmarking pipeline for dataset BrainInvaders15a with Twin NN

name: "BrainInvaders2015a"

dataset:
- name: BrainInvaders2015a
  from: neuroIDBench.datasets
  parameters:
  - subjects: 10
  - interval: [-0.1, 0.9]
  - rejection_threshold: 200

pipelines:
"TNN":
- name: TwinNeuralNetwork
  from: neuroIDBench.featureExtraction
  parameters:
  - EPOCHS: 10
  - batch_size: 256
  - verbose: 1
  - workers: 1

"AR+PSD+SVM":
- name: AutoRegressive
  from: neuroIDBench.featureExtraction
  parameters:
  - order: 6
- name: PowerSpectralDensity
  from: neuroIDBench.featureExtraction
- name: SVC
  from: sklearn.svm
  parameters:
  - kernel: 'rbf'
  - class_weight: "balanced"
  - probability: True

Listing 5: Benchmarking pipeline for dataset BrainInvaders15a with shallow classifiers and twin NN

name: "BrainInvaders2015a"

dataset:
- name: BrainInvaders2015a
  from: neuroIDBench.datasets
  parameters:
  - subjects: 10
  - interval: [-0.1, 0.9]
  - rejection_threshold: 200

pipelines:
"TNN":
- name: TwinNeuralNetwork
  from: neuroIDBench.featureExtraction
  parameters:
  - EPOCHS: 10
  - batch_size: 256
  - verbose: 1
  - workers: 1

"AR+PSD+SVM":
- name: AutoRegressive
  from: neuroIDBench.featureExtraction
  parameters:
  - order: 6
- name: PowerSpectralDensity
  from: neuroIDBench.featureExtraction
- name: SVC
  from: sklearn.svm
  parameters:
  - kernel: 'rbf'
  - class_weight: "balanced"
  - probability: True

Listing 6: Benchmarking pipeline for multi-session dataset COGBCI: FLANKER with shallow classifiers and Twin NN

name: "COGBCIFLANKER"

dataset:
- name: COGBCIFLANKER
  from: neuroIDBench.datasets
  parameters:
  - subjects: 10
  - interval: [-0.1, 0.9]
rejection_threshold: 200

pipelines:
  "TNN":
    - name: TwinNeuralNetwork
      from: neuroIDBench.featureExtraction
      parameters:
        EPOCHS: 10
        batch_size: 256
        verbose: 1
        workers: 1

  "AR+PSD+RF":
    - name: AutoRegressive
      from: neuroIDBench.featureExtraction
      parameters:
        order: 6
        class_weight: "balanced"

  "AR+PSD+SVM":
    - name: AutoRegressive
      from: neuroIDBench.featureExtraction
      parameters:
        order: 6
        class_weight: "balanced"

    - name: PowerSpectralDensity
      from: neuroIDBench.featureExtraction
      parameters:
        kernel: 'rbf'
        class_weight: "balanced"
        probability: True

    - name: SVC
      from: sklearn.svm
      parameters:
        kernel: 'rbf'
        class_weight: "balanced"
        probability: True

Listing 7: Benchmarking pipeline for User i.e., Reseracher's own MNE data with shallow classifiers and Twin NN

name: "User"

dataset:
  - name: UserDataset
    from: neuroIDBench.datasets
    parameters:
      dataset_path: <local_dataset_path>

pipelines:
  "AR+PSD+SVM":
    - name: AutoRegressive
      from: neuroIDBench.featureExtraction
      parameters:
        order: 6
        class_weight: "balanced"

    - name: PowerSpectralDensity
      from: neuroIDBench.featureExtraction
      parameters:
        kernel: 'rbf'
        class_weight: "balanced"
        probability: True

    - name: SVC
      from: sklearn.svm
      parameters:
        kernel: 'rbf'
        class_weight: "balanced"
        probability: True

Listing 8: Benchmarking pipeline for User i.e., Reseracher's own EEG data and Researcher's own customized method for Twin NN

name: "User"

dataset:
  - name: UserDataset
    from: neuroIDBench.datasets
    parameters:
      dataset_path: <local_dataset_path>

pipelines:
  "TNN":
    - name: TwinNeuralNetwork
      from: neuroIDBench.featureExtraction
      parameters:
        user_tnn_path: <local_path_to_TNN>
        EPOCHS: 10
        batch_size: 256
        verbose: 1
        workers: 1

B.3. Automated benchmarking

After setting the parameters for the benchmarking for single dataset. The automated benchmarking can be performed either by running the automation script from the command line terminal or by running the script within the Docker container.

Running the tool using command line terminal

Execute the following command from the root directory:

```
python brainModels/run.py
```

Running the tool using Docker

Open the docker desktop installed on our local machine and execute the docker image my_dockerfile.

References

A.K. Chaurasia et al.


