Towards Mobile-Based Preprocessing Pipeline for Electroencephalography (EEG) Analyses: The Case of Tinnitus

Muntazir Mehdi[1(✉)], Lukas Hennig[1], Florian Diemer[1], Albi Dode[2], Rüdiger Pryss[3], Winfried Schlee[4], Manfred Reichert[2], and Franz J. Hauck[1]

[1] Institute of Distributed Systems, Ulm University, Ulm, Germany
{muntazir.mehdi,lukas.hennig,florian.diemer,franz.hauck}@uni-ulm.de
[2] Institute of Databases and Information Systems, Ulm University, Ulm, Germany
[3] Institute of Clinical Epidemiology and Biometry, University of Würzburg, Würzburg, Germany
[4] Clinic and Policlinic for Psychiatry and Psychotherapy, Regensburg, Germany

Abstract. Recent developments in Brain-Computer Interfaces (BCI)—technologies to collect brain imaging data—allow recording of Electroencephalography (EEG) data outside of a laboratory setting by means of mobile EEG systems. Brain imaging has been pivotal in understanding the neurobiological correlates of human behavior in many complex disorders. This is also the case for tinnitus, a disorder that causes phantom noise sensations in the ears in absence of any sound source. As studies have shown that tinnitus is also influenced by complexities in non-auditory brain areas, mobile EEG can be a viable solution in better understanding the influencing factors causing tinnitus. Mobile EEG will become even more useful, if real-time EEG analysis in mobile experimental environments is enabled, e.g., as an immediate feedback to physicians and patients or in undeveloped areas where a laboratory setup is unfeasible. The volume and complexity of brain imaging data have made preprocessing a pertinent step in the process of analysis, e.g., for data cleaning and artifact removal. We introduce the first smartphone-based preprocessing pipeline for real-time EEG analysis. More specifically, we present a mobile app with a rudimentary EEG preprocessing pipeline and evaluate the app and its resource consumption underpinning the feasibility of smartphones for EEG preprocessing. Our proposed approach will allow researchers to collect brain imaging data of tinnitus and other patients in real-world environments and everyday situations, thereby collecting evidence for previously unknown facts about tinnitus and other conditions.

Keywords: Healthcare · Mobile health · Smartphone apps · Mobile apps · Tinnitus · EEG · Brain imaging · Brain Computer Interfaces

J. Ye et al. (Eds.): MobiHealth 2020, LNICST 362, pp. 67–86, 2021.
https://doi.org/10.1007/978-3-030-70569-5_5

1 Introduction

Brain imaging techniques offer different opportunities to examine the neurobiological correlates of human behavior. Among different brain-imaging techniques, for instance, Magnetoencephalography (MEG), Functional Magnetic Resonance Imaging (fMRI), and Positron Emission Tomography (PET), Electroencephalography (EEG) is the most adaptable and multifaceted one. EEG is a non-invasive tool that allows the investigation of the resting-state electrical activity of the brain by means of electrodes positioned on the scalp [8]. This enables the investigation of human brain functions by recording the communication between neurons in the brain network measured in volts. EEG offers high time resolution (high number of snapshots of electrical activity from various electrodes) in comparison to fMRI and PET [25], and is an inexpensive and low maintenance technique compared to MEG. Thus, EEG is not only an inexpensive but a versatile, lightweight, and portable brain-imaging technique, and it is extensively applied in tinnitus research [3,7,11].

Tinnitus is a common disorder responsible for causing the perception of a ringing sound in the ears without presence of any external sound source. The reasons pertaining to causing this phantom sound are yet to be fully discovered, but it has been firmly established that tinnitus is caused by an underlying anomaly in the ear such as damage and loss of cochlear hair cells [14]. Despite the fact that tinnitus is traditionally considered a problem of the inner ear, recent studies using brain imaging have shown that the complexity of tinnitus goes beyond the auditory cortex into non-auditory brain areas [8,13]. Brain imaging techniques like EEG can be pivotal in collecting evidences for further yet unknown facts regarding the neuronal activity of tinnitus.

Current developments in EEG research have progressed significantly to record EEG outside a laboratory setting by means of ambulatory or mobile EEGs [16,17]. Mobile EEG devices are equipped with necessary hardware to be communicated by a wired (USB) or a wireless connection (Bluetooth or WiFi). Generally, an EEG session is primarily recorded and temporarily stored on the mobile EEG device (either on the built-in flash memory or an external SD card). Since brain imaging outside a controlled laboratory setting and in real-world scenario can result in unnecessary noise in data and useless subject-generated artifacts [12], the EEG recordings are therefore transferred to a computer for preprocessing steps like data cleansing, filtering and artifact removal using EEGLab Scripts [6], MATLAB, or FieldTrip [23]. Alternatively, mobile EEG can also be directly connected to a computer to transfer real-time EEG data and perform on-the-run preprocessing [26].

Although the current paradigm of EEG recording and preprocessing is a significant improvement over conventional EEG, including EEG analysis in real-world settings, it is still limited in terms of offering real-time analysis with freedom of movement or mobility. A major shortcoming of the current EEG analysis paradigm is the requirement of additional hardware for EEG data acquisition, preprocessing, and visualization. For example, currently, the overall process of EEG analysis and visualization requires additional steps of transferring EEG

data to a computer, thereby hindering the mobility and introduction of requiring specialized software for EEG data preprocessing. A possible alternative solution to this problem can be *Mobile Sensing*—the process of acquiring sensory data of an individual using a smartphone or mobile device while allowing mobility [20]. Smartphones are capable to be used, and some scientific literature has already reported their successful usage [19,30].

Modern smartphones are ubiquitous devices that provide sophisticated communication hardware, exceptional computing power, and reasonable battery. Additionally, smartphones offer APIs for programming new apps. These characteristics plus the fact that smartphones are literally mobile devices make smartphones an ideal candidate for real-time analysis and recording of EEG data in non-conventional, exceptional, and atypical real-world settings such as swimming, running, or hiking etc. However, it is also notable that the smartphones are manufactured as general purpose devices and are not specialized for real-time EEG recording and analysis, therefore, their feasibility and behavior in such cases require efforts. For instance, continuous sampling of the EEG data might result in excessive battery consumption problems [33], or might introduce scarcity of computational power for general user experience [2]. Furthermore, a continuous Bluetooth connection with the mobile EEG device might cause data transmission problems [10], as well as its associated energy consumption problem [32].

Therefore, for addressing the aforementioned challenges, this article proposes a mobile-based preprocessing pipeline for EEG analysis, more specifically (i) the development and design of a smartphone app with a rudimentary EEG preprocessing pipeline, and (ii) an evaluation of the proposed app to show the feasibility of smartphones to perform EEG preprocessing. The proposed work is motivated and driven by the needs of tinnitus research within the context of the European School for Interdisciplinary Tinnitus Research (ESIT) [29]. One core goal of the ESIT project is the development of a generic, robust and flexible middleware for mobile crowdsensing to monitor real-time measurements of tinnitus-related parameters as well as electroencephalographic and physical activities. The proposed approach will improve mobility for EEG data acquisition and analysis using smartphones and enable preprocessing of EEG data without the need of specialized software and hardware. The proposed smartphone app will also allow researchers to collect brain-imaging data of tinnitus patients in a variety of experimental conditions in real-world environment, thereby, to collect evidence for unknown facts regarding tinnitus in brain regions. In particular, the proposed smartphone app will assist researchers in designing and gathering EEG data for large scale longitudinal studies, for example, to investigate oscillatory brain activity of tinnitus patients in a longitudinal design by investigating patients that have moments with high and low tinnitus intensity. Furthermore, the ability to collect and analyze real-time EEG data in real-world experimental situations as well as in places where a laboratory EEG setup is impossible—for instance, in underdeveloped or undeveloped rural areas—will be a significant asset for brain-imaging and neuro-imaging research. The application possibilities

are not limited to tinnitus research, but the proposed solution will also support a variety of application domains where brain-imaging is vital.

Section 2 of this paper gives insights into previously reported related work in this field and briefly discusses the existing preprocessing approaches. Section 3 details the overall design and implementation of the proposed work. The subsequent Sect. 4 evaluates the proposed approach by presenting results and data on the feasibility of smartphones for preprocessing EEG data. Finally, we conclude and present brief insights into future work in Sect. 5.

2 Related Work

In terms of specialized software packages for offline and online preprocessing and analysis of EEG data, EEGLAB [6] and FieldTrip [23] are among the most prominent. EEGLAB is an open source (GNU license) toolbox for MATLAB. It is used for processing EEG data, including data filtering and artifact removal, as well as analysis of EEG data using Independent Component Analysis (ICA). Similarly, FieldTrip is also an open source (GNU license) toolbox for MATLAB for analyzing EEG data. In terms of developing BCI applications, OpenViBE (framework for developing BCI applications for neurofeedback and biofeedback) [27], BCILAB (EEGLAB plugin to develop EEG predictive models) [15], and BCI2000 (a C++ framework for developing real-time BCI applications) [28] are some of the popular frameworks. Furthermore, Esch et al. [9] present the MNE software project, which comprises tools required for EEG and MEG data acquisition, preprocessing, analysis, and visualization. Similarly, Tadel et al. [31] present an open-source platform for EEG and MEG data analysis and visualization.

With reference to existing preprocessing pipelines, it is pertinent to notice that there exists no standard method. Usually, the preprocessing of EEG signals is supervised by EEG experts. However, there has been some existing literature reporting on automated preprocessing of EEG data. Usually, most of the pre-existing preprocessing pipelines perform filtering, removal of line noise, and detection of bad channels including interpolation. Among the preexisting preprocessing pipelines, the PREP Pipeline [4] claims to standardize the preprocessing of EEG data. The main idea of PREP is to distinguish externally generated noise, such as electrical interference and patient-generated artifacts via muscular activation. For instance, the line-noise detection and removal is done using a modified implementation of the *CleanLine* plugin from EEGLAB [1,22]. The PREP Pipeline has been reused in other preprocessing implementations, Automagic [24] and the Batch Electroencephalography Automated Processing Platform (BEAPP) [18]. In [5], da Cruz et al. propose a MATLAB-based automated preprocessing pipeline for EEG data called APP. APP uses the CleanLine plugin from EEGLAB for line-noise removal like the PREP pipeline. Furthermore, APP applies a *3rd Order Butterworth filter* 1 Hz in both forward and reverse direction to correct the direct-current (DC) drift caused by changes in the DC value. After removing the line noises, the channel data is re-referenced. Both PREP and the APP preprocessing pipelines extensively use the EEGLAB preprocessing library. Instructions on how to preprocess EEG data using EEGLAB

Table 1. Overview of the preprocessing pipelines

PREP [4]	APP [5]	Makoto [21]	Result
–	3rd order Butter-worth filter	Highpass filter	3rd order Butter-worth filter
Cleanline	CleanLine	CleanLine	Band-stop filter
Signal true mean estimation with bad channels interpolated	Signal true mean estimation with weighted mean	–	Estimate signal true mean with bad channels interpolated
Detect bad channels relative to mean and interpolate	Detect bad channels relative to neighbors and with high dis-persion to mean	–	Detect bad channels relative to mean and interpolate
Detecting noisy or outlier channels	–	–	–
–	Detecting and remove bad epochs	Reject epochs for cleaning	–
–	ICA	ICA	–
–	Detection, removal and interpolation of bad channels in epochs	–	–
–	Outlier detection	–	–

and development of preprocessing pipelines are given by Makoto Miyakoshi from Swartz Center for Computational Neuroscience [21].

The three foremost and commonly used preprocessing pipelines (PREP, APP, Makoto) are delineated in Table 1, along-with a comparison to our proposed approach. We first apply a *3rd Order Butterworth filter* in both forward and reverse directions for signal filtering like the APP preprocessing pipeline. Next, we use a *Band-stop Filter* (also called notch filter) as an alternative to the Cleanline to remove power line interference between 50 and 60 Hz or 50 and 70 Hz. Despite that this can cause significant signal distortion around the band-stop frequency and phase distortion [4], however, our choice of implementing band-stop filter is due to resource scarcity on the smartphones. Currently, we are working on implementing and optimizing the CleanLine algorithm for the Android platform. Finally, in order to detect bad channels, we have implemented

and modified both phases of the PREP's 'Referencing Procedure' for the Android platform.

In general, there exists a plethora of literature reporting on software packages and toolboxes for online and offline analysis of EEG data. Similarly, there exist plenty of literature reporting on automated preprocessing pipelines and standardizing the preprocessing of EEG data. Our literature review did not yield any study that reports on any application of preprocessing EEG data using smartphones. Specifically, we did not find any article that benchmarks the preprocessing of EEG data using smartphones. To the best of our knowledge, the proposed work is the first of its type towards mobile-based preprocessing pipeline for EEG analysis, including visualization of EEG data, as well as to present evidence regarding feasibility of smartphones to perform preprocessing of EEG data.

3 Implementation

The proposed work aims at preprocessing of EEG data for analysis purposes using a smartphone. Therefore, we have developed an Android application. The overall architecture of the proposed app is presented in Fig. 1. The data from electrodes of the EEG cap are transmitted to the EEG Amplifier. In our implementation, we have used the EEG Amplifier by Brain Products called LiveAmp 16[1]. The EEG Amplifier can be coupled with the smartphone using Bluetooth.

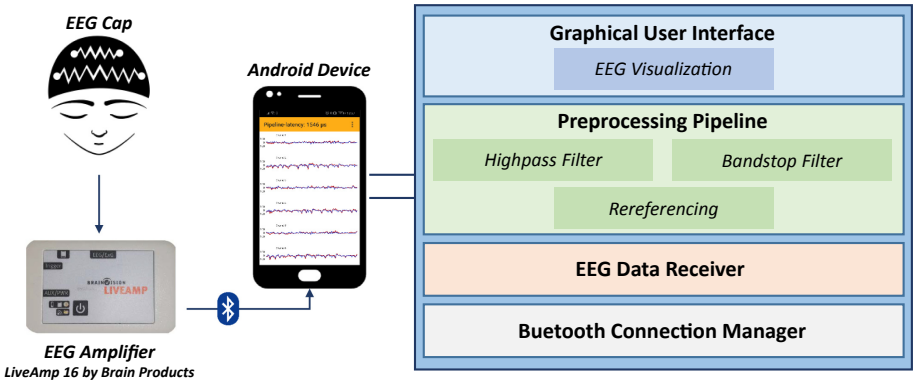


Fig. 1. Architecture

To acquire live EEG data, an EEG cap is connected to LiveAmp 16 using a wired connection. The *Bluetooth Connection Manager* module is responsible for establishing the first connection with LiveAmp 16, and maintaining the

[1] https://www.brainproducts.com/productdetails.php?id=63 Accessed: 15/06/2020.

Bluetooth connection for the duration of EEG. The *EEG Data Receiver* module is implemented in Java and included as an external library to the Android application. The EEG data-receiver module is responsible for communicating with the LiveAmp 16 based on the LiveAmp-16's communication protocol, and is assisted by the Bluetooth-connection manager module. All communications between LiveAmp 16 and the Android application are done in a proprietary binary data format using request-response method. Some examples of requests sent to LiveAmp 16 are 'Get Device Information' and 'Get Device Status'. In order to start EEG data acquisition, the EEG data receiver sends a request of type 'Start Data Acquisition' and starts receiving EEG data in binary responses from LiveAmp 16. The data transformation is also managed by the EEG data-receiver module. Once the data has been transformed into an internal Java format, the EEG data is forwarded to the *Preprocessing-Pipeline* module.

3.1 Preprocessing Pipeline

Our current implementation of the preprocessing pipeline offers filtering, removal of line noise, and detection of bad channels including interpolation. As these steps are usually part of any preprocessing pipeline, we identify these steps to be principle components, and therefore we have limited our current implementation to these. Herein, the filtering is offered by `HighPassFilter` with a *3rd Order Butterworth* filter. The line noise removal is carried out by `BandStopFilter`. Finally, the bad-channel detection is done by adopting and implementing both phases of the PREP's 'Referencing Procedure' for Android platform, we refer to as `Rereferencer`. The overall design of our current implementation of the preprocessing pipeline is illustrated in Fig. 2, using a Class Diagram, and the sequential object interactions of the Java classes is given in Fig. 3. The individual classes as well as their relations are briefly discussed below:

Pipeline. The abstract `Pipeline` class defines all the necessary properties and methods, such as the frequencies of filters, and the sampling rate. The `Pipeline` class implements the `Filter` interface, where the `Filter` interface declares the method called `filter()`. To ensure a flexible class design and a uniform filter structure, all filter classes extend the abstract `Pipeline` class (`HighPassFilter`, `Rereferencer`, and `BandStopFilter`). The filter-specific logic is implemented in the overridden `filter()` method of each extending filter class. In addition to other properties, the `Pipeline` class also defines an instance of `EpochBuffer` class. During the filtering process, the `Pipeline` class initializes the buffer with the EEG data values.

EpochBuffer. To work with continuously incoming data, an `EpochBuffer` with a default length of 64 values is used. The `EpochBuffer` implements a `CircularBuffer` and stores the incoming EEG data values. All filters are sequentially applied on the values stored in buffer, thus modifying the EEG data one after the other.

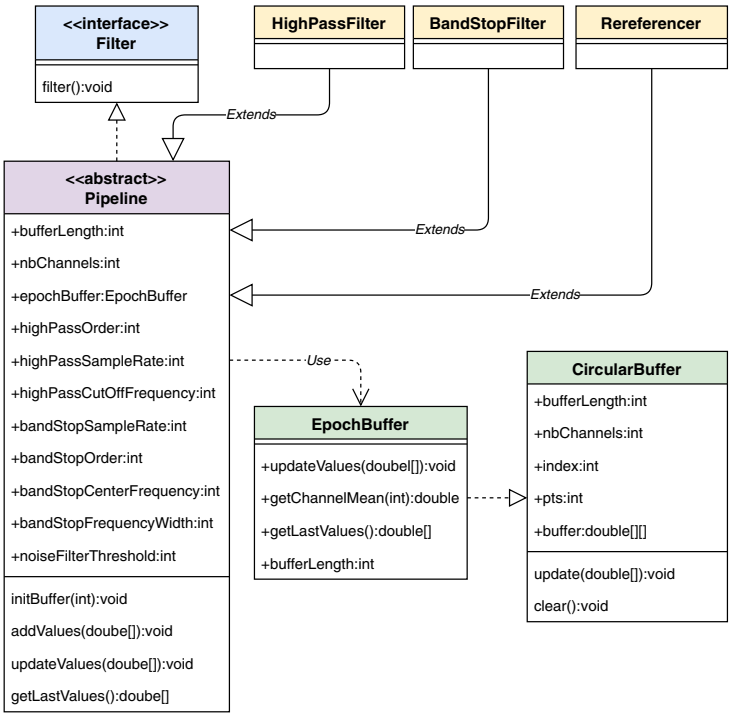


Fig. 2. Class diagram of preprocessing pipeline-related Java classes

HighPassFilter. Class `HighPassFilter` is implemented with the help of an Infinite Impulse Response (IIR) filter library for digital signal processing[2]. The library is integrated into the project using Maven. The library allows application of 3rd Order Butterworth Filter with a default value 1 Hz and a sampling rate 250 Hz to the signal.

Rereferencer. For this filter, both phases of the PREP's re-referencing algorithm presented in [4] was implemented in Java for the Android platform. The `NoiseDetector` from NeuroTechX[3] used in EEG-101 was used to detect noisy channels. The noise detector uses variance thresholding on the data available in `EpochBuffer` to detect and mark noisy channels.

BandStopFilter. The aforementioned IIR library comes with an implementation of the band-stop filter. To remove line noise from the signal, our implementation re-uses the band-stop filter from the IIR library.

[2] https://github.com/berndporr/iirj Accessed: 15/06/2020.
[3] https://github.com/NeuroTechX/eeg-101 Accessed: 15/06/2020.

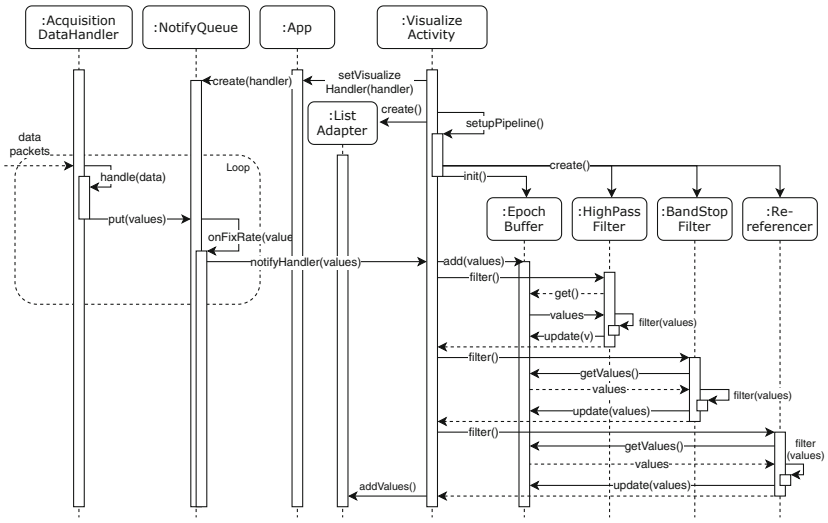


Fig. 3. Sequence diagram of data preprocessing

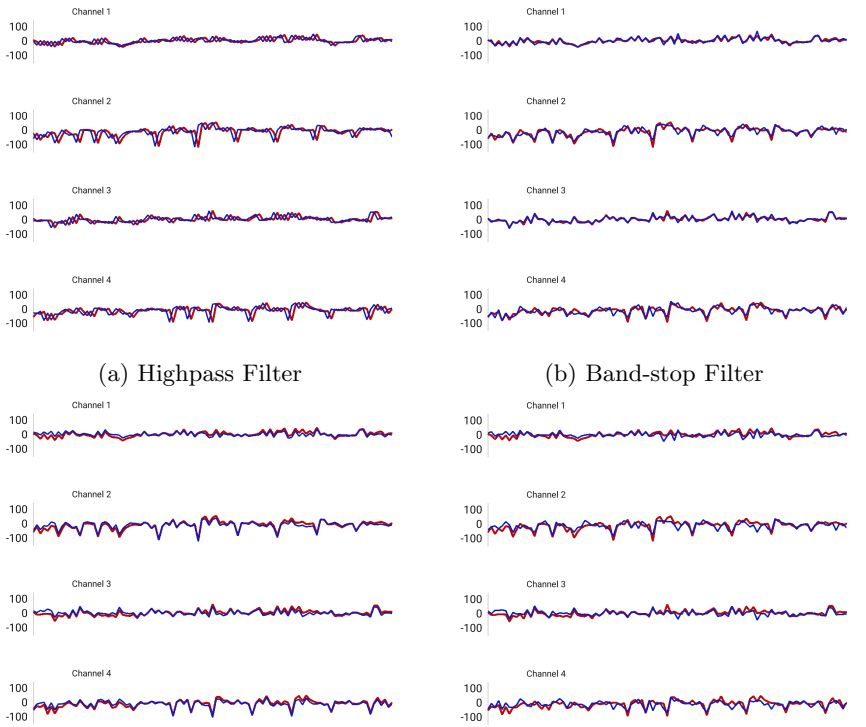
To better examine the results and behavior of the individual filters, as well as the entire pipeline (all filters applied), a comparison of filter application to the raw EEG data is shown in Fig. 4. In all represented graphs in Fig. 4(a–d), the blue signal represents the raw EEG data without application of any filter, while the blue signal represents the EEG data after application of individual filter. Herein, Fig. 4a shows the comparison of raw EEG data and the high-pass filter, Fig. 4b shows a comparison of the band-stop filter with raw EEG data, and Fig. 4c shows a comparison of raw EEG data and the application of the re-referencer filter. Similarly, Fig. 4d shows a comparison between raw EEG data with all filters applied (high pass, band-stop, re-referencer).

From Fig. 4a, we can observe very minor difference between the two signals, suggesting very little impact on changing the signal. Figure 4b gives a good example of influence of the band-stop filter on the EEG data. Although the signal looks quite similar to the original, but at some points the peaks become more smoother. With the re-referencer filter results shown in Fig. 4c, it can be noticed that the signal peaks remain in their amplitude, but in some places there is a slight upward and downward shift in amplitude of the signal, particularly in the signal comparison of Channel 1. The result of the entire pipeline, depicted in Fig. 4d, shows a mixture of what we experienced at each individual filter.

The exact accuracy of application of individual filters can be questioned, therefore, domain experts can be helpful in validating and improving the filter implementation. Furthermore, please also note that the amount of influence of applying individual filters as well as the entire pipeline on the EEG data is dependent and subjective of the type of raw EEG data used. For instance, a cleaner input EEG signal with minimum noise and noisy artifacts will present

minimum change in the output EEG signal after application of the pipeline filters.

It is also pertinent to note here that, even if the filters are cleaning the EEG data, the inclusion of experts is necessary to clarify whether the resulting EEG data after application of the pipeline filters does not contain any noisy artifacts. Similarly, domain experts can also advise in case if the filters are responsible for removing any significant information from the input EEG data, which is critical for the domain-specific analysis. In both of these cases, respective filter parameters can be modified and adjusted to find an optimal filter setting.



(a) Highpass Filter (b) Band-stop Filter

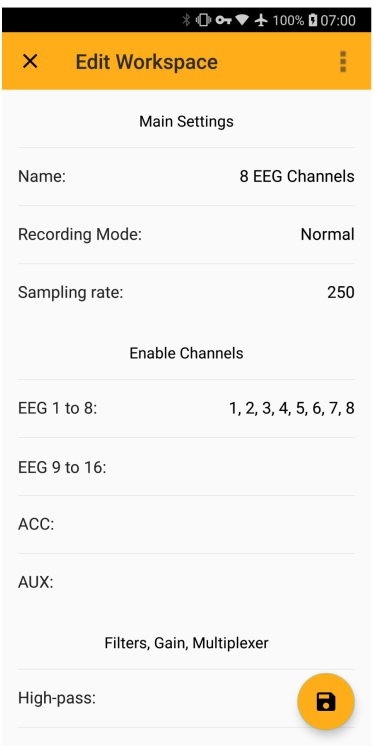
(c) Re-referencer (d) Pipeline

Fig. 4. Comparison between raw simulator data (red) and the filters applied (blue) (Color figure online)

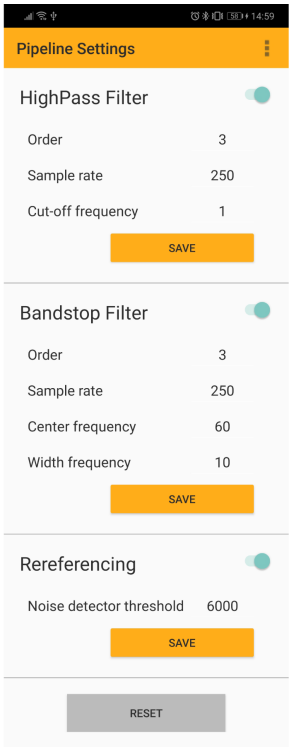
3.2 Graphical User Interface (GUI)

Workspace and Filter Settings

Before running an EEG recording session, the EEG device must be configured properly. The proposed Android application uses workspaces for this task.



(a) EEG workspace creation



(b) Preprocessing filter settings

Fig. 5. EEG application screenshots

Workspaces are stored and can be edited later. This allows the flexibility to execute multiple EEG sessions with the same workspace configuration. Additionally, changing a single parameter of an existing workspace is also possible. Figure 5a shows the screenshot from the app for the workspace creation. Each workspace consists of several parameter settings like name, recording mode, and sampling rate. Furthermore, the workspace screen also allows enabling and disabling of EEG channels. The workspace configuration can be stored on the Android device and are sent to the EEG amplifier before an EEG session via *EEG Data Receiver* module.

In addition to the workspace configuration, the proposed Android application also allows configuration of filters through a *Pipeline Settings* screen. The pipeline-settings screen allows enabling and disabling of individual pipeline filters as well as configuration of filter parameters. This allows the behavior of individual filters, or different combination of filters on the EEG data to be observed and evaluated. Additionally, changing configuration parameters of individual filters

allows optimization of filter application on the EEG signal. A screenshot from the Android app for pipeline settings is depicted in Fig. 5b.

Visualization

In order to visualize EEG data (prior or post-preprocessing), we implemented an Android-specific `ListAdapter`. This `ListAdapter` in an integral part of the aforementioned *EEG data receiver* module, and is responsible for establishing and managing data communication between the data model and visualization. The overall structure of `ListAdapter` is depicted in Fig. 6. The *Data Model* component is implemented as Java POJO Classes to hold specific data of EEG channels. As there is a lack of native Android chart libraries, the MPAndroidChart library by Philipp Jahoda[4] is used for creation of line charts to show EEG data. Since not all channels should be displayed in a single chart, the `ListAdapter` provides a `ViewHolder` for each and individual EEG channel using a line chart. Once an EEG data packet of all channels has been preprocessed through the pipeline, it is forwarded to the list of channels in the `ListAdapter`. The adapter is then informed by the `Notifier` component (Java listener component triggered on changes in EEG data packet values) that its list of channels has new data values and can therefore update the `ViewHolder`. The `ViewHolder` holds the line charts and updates them with each new EEG data packet. Since data outside the Android viewport is invisible and is irrelevant for display, therefore, the number of data values in individual line chart is limited to the viewport, this allows conservation of the working memory of the smartphone. In order to further conserve the smartphone resource, the `RecyclerView` component ensures refreshing of `ViewHolder` based on last used EEG data packets in case the EEG data packet values have not changed.

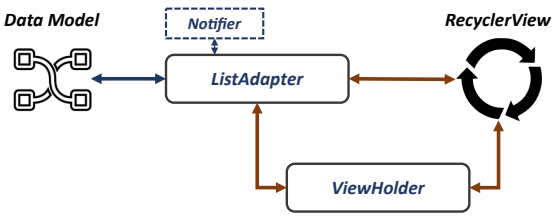
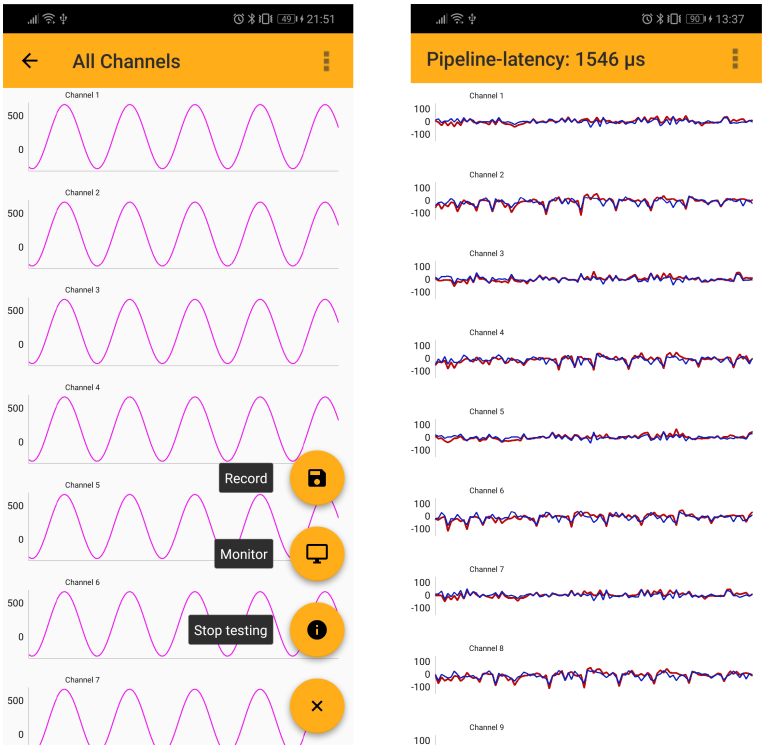


Fig. 6. Structure of the ListAdapter for visualization of EEG data

Figure 7a shows an example visualization of the test signal generated by the EEG amplifier device for Channels 1 to 8. Furthermore, please note the control buttons on the bottom right corner of the screen. The control buttons are divided into three types: 1) The Record button starts recording (storage of EEG data on smartphone and amplifier) of the EEG along with visualization of the EEG

[4] https://github.com/PhilJay/MPAndroidChart Accessed: 15/06/2020.

signal on the smartphone, 2) the Monitor button starts visualizing the EEG data without recording, and 3) Start/Stop testing starts respectively stops receiving test signal (sinusoidal wave) generated by the amplifier to test connectivity and data transmission. Figure 7b shows an example visualization of a real EEG data. The pipeline latency on the top of the screen shows the time delay between the preprocessing of two consecutive EEG data packet values. In this example, the latency is shown for all three pipeline filters.



(a) Test Signal Visualization (b) EEG Data Visualization

Fig. 7. EEG application screenshots

4 Results and Discussions

Since one of the core goals of our proposed work was to test and evaluate the feasibility of smartphones for EEG data preprocessing, in this section we detail the experiments and results examining the performance of the proposed preprocessing pipeline on a mobile device. We have exhaustively tested our proposed approach and run experiments to provide a detailed comparison of resource consumption on the mobile device for acquiring raw EEG data (non-processed EEG

data), application of individual filters on the EEG data, and application of all filters on the EEG data. Our experiments focus specifically on mobile resource consumption in terms of CPU usage, working memory usage, and battery consumption.

4.1 Experimental Setup

To measure the performance data of the proposed pipeline and its filters, the Huawei P20 Lite with 4 GB RAM, with an Octacore processor Kirin 659 (4×2.36 GHz + 4×1.7 GHz), and non-removable Li-Po 3000 mAh battery was used[5]. The *Android Profiler* built into Android Studio was used to measure the app performance. The workload and resource consumption of raw data, individual filters and entire preprocessing pipeline were captured by running them for a duration of 5 min. The entire process was repeated 3 times, the performance data was recorded, and the arithmetic mean of 3 separate runs was computed. The EEG amplifier configurations and filter settings used for the experiments are given in Table 2.

Table 2. EEG amplifier and filter settings

Settings	Type	Values
Workspace	EEG channels	1–8
	Data type	Test
	Sampling rate	250
HighPass filter	Order	3
	Cut-off frequency	1
BandStop filter	Order	3
	Center frequency	60
	Width frequency	10
Rereferencing	Variance threshold	4000

4.2 Results

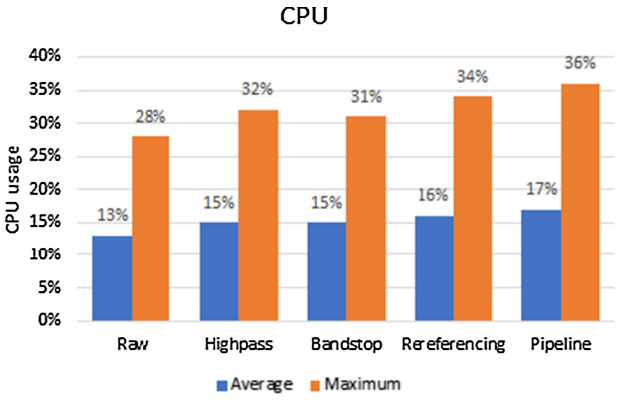
The comparative performance results of the proposed EEG preprocessing pipeline are given in Fig. 8, where Fig. 8a gives performance in terms of CPU usage in percentage, and Fig. 8b shows the amount of working memory used in MB. The battery related results are shown in Fig. 9, where Fig. 9a shows the energy consumption in percentage. Herein, please note that the Android Studio Profiler only distinguishes between three energy levels namely light, medium, and heavy. We divided each of those levels into three equal parts which results

[5] https://consumer.huawei.com/de/support/phones/p20-lite/ Accessed: 15/06/2020.

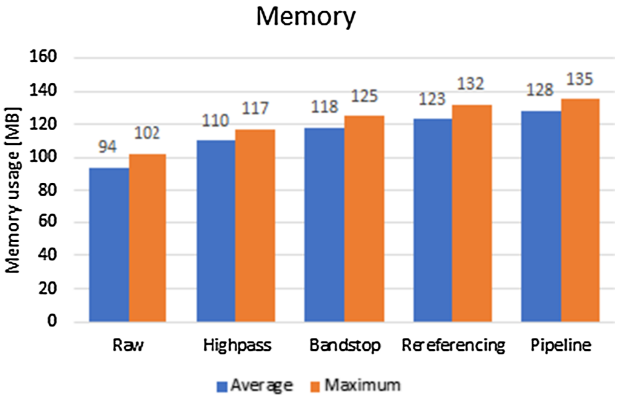
in nine same sized intervals. The introduced nine intervals were used for proper quantization of the energy used by filters and allow better distinction of energy consumption. Figure 9b shows the comparative results of over-all battery runtime duration in hours (hh:mm format). For this purpose, the mobile device was completely charged and the EEG data was continuously sampled, processed, and visualized until the battery was exhausted.

4.3 Discussion

From Fig. 8, in general, we can see minimal usage of critical computing resources of the smartphone. Note that this is suggestive as the regular user experience, including the background services, can not be hindered by the preprocessing and visualization of EEG data. Specifically, from Fig. 8a, the average CPU

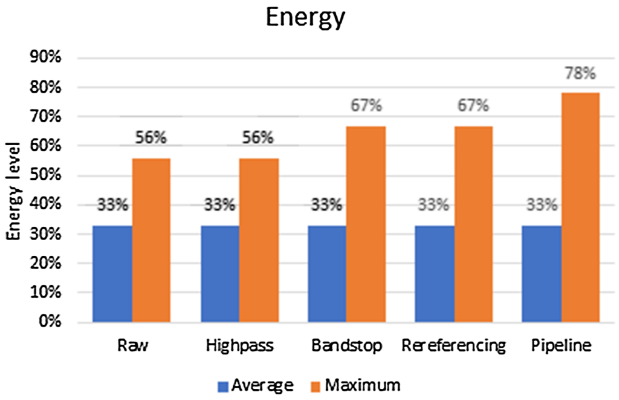


(a) CPU load in percentage

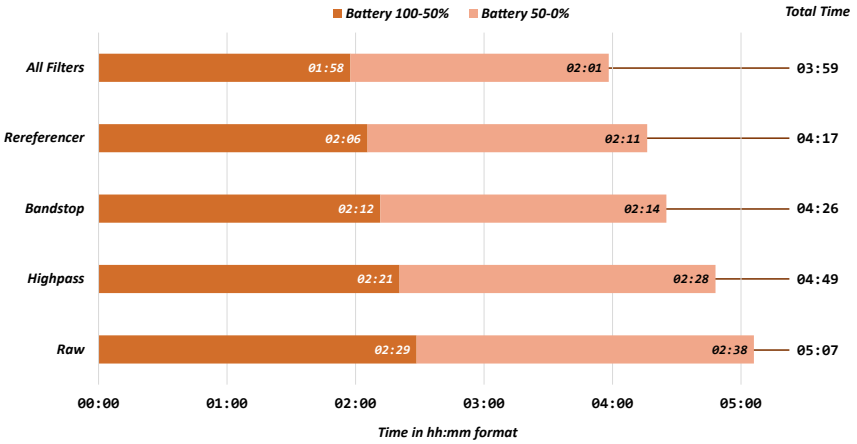


(b) Memory consumption in MB

Fig. 8. Smartphone performance results



(a) Energy Consumption



(b) Battery run-time durations

Fig. 9. Smartphone battery results

usage ranges between 13%–17%, with highest usage by the entire preprocessing pipeline. In case of maximum CPU usage, we see varying values between 28%–36% and highest consumption of 36% by the entire pipeline. In comparison to acquiring and visualizing raw EEG data, the amount of extra CPU usage (CPU overhead) by applying the entire pipeline is notably lower (average CPU usage difference of 4% and maximum CPU usage difference of 8%). In case of both average and maximum CPU usage, the values for each individual filters remained on the same level with slight difference in comparison to the raw EEG data CPU usage values.

From Fig. 8b, very nominal amounts of working memory or RAM usage can be seen. The average memory usage ranges between 94–128 MB, with most memory usage of 128 MB by the entire preprocessing pipeline. In case of maximum memory usage, a variation of values ranging between 102–135 MB were

observed, where the highest memory usage of 135 MB was for the entire pipeline. Comparatively, the amount of additional working memory required for applying the entire pipeline as opposed to acquiring and visualizing raw EEG data is very low (average memory usage difference of 34 MB and maximum memory usage difference of 33 MB).

Apparently, the proposed preprocessing pipeline is resource-intensive in terms of battery and energy consumption (see Fig. 9). A moderate to high requirement for energy consumption was already anticipated due to the involvement of additional resources like Bluetooth and the smartphone screen usage. However, from Fig. 9a, we can conclude an acceptable energy requirement by the app. Specifically, since in case of average energy usage, all filters as well the entire preprocessing pipeline consumed 33% of the smartphone energy, inline with the battery usage for acquiring and visualizing raw EEG data. Conversely, in case of maximum energy usage, we see values ranging from 56% (energy usage to acquire, and visualize raw EEG data) to 78% (energy usage for acquiring, preprocessing with entire pipeline, and visualizing raw EEG data). Herein, we see a notable additional energy usage of 22% by the entire preprocessing pipeline. For `HighPass` filter, the energy remains same as the raw EEG data acquisition and visualization, but an additional energy usage of 11% for band-stop and referencing (`Rereferencer`) filters can be seen.

On the other hand, energy usage measure can be subjective in certain scenarios, therefore, an objective measure in terms of overall battery run-time duration is given in Fig. 9b. The overall battery run-time duration represents the amount of time between a full battery charge and empty battery. From Fig. 9b, we see a total of 5 h and 7 min alive time for continuous raw EEG data acquisition and visualization. The overall time duration varied for EEG data acquisition, visualization, and applying individual preprocessing filter. For instance, for Highpass filter the battery lasted for 4 h 49 min, for bandstop filter the battery run-time duration was 4 h 26 min, 4 h 17 min for the Rereferencer, and for the entire pipeline (all filters) the battery lasted for 3 h and 59 min. Herein, we see the lowest battery run-time duration of 3 h and 59 min, which is acceptable since most conventional EEG sessions require maximum of 40 min.

5 Conclusion, Limitations, and Future Work

Portable, ambulatory, or mobile EEG devices allow monitoring of neuronal activities of human brain in real-life scenarios. The mobile EEG devices support the wireless transmission of EEG data over Bluetooth, thus, enabling live EEG data processing and visualization on standard smartphones. In this work, we proposed an elementary mobile-based preprocessing pipeline for EEG analysis and evaluated the feasibility of smartphones for EEG data preprocessing. Our experiments and results show that contemporary smartphones have satisfactory computational capabilities in terms of CPU and working memory to perform EEG data acquisition, preprocessing, and visualization without hindering the user-experience in relation to general smartphone use. Further, our experiments

with battery consumption while preprocessing and visualizing EEG data show moderate energy consumption and suggest that the smartphones hold ample battery capacity to allow recording of multiple EEG sessions. The proposed approach was realized within the context of tinnitus research to collect evidence for unknown facts regarding tinnitus using brain imaging techniques. The significant contributions of the proposed approach are to, (i) improve EEG data acquisition, preprocessing, visualization, and analysis, (ii) enable preprocessing of EEG data using smartphones and without the need of specialized software or hardware, (iii) allow researchers the flexibility to gather brain imaging data of tinnitus patients in a variety of experimental conditions in real-world environments. Furthermore, the proposed app serves as an initial step towards smartphone-based automated mobile neurofeedback and biofeedback for tinnitus patients. Nevertheless, our approach can be applied for other domains needing mobile and real-time EEG observations.

Two notable shortcomings of our proposed work are, 1) the number of filters included in the preprocessing pipeline, and 2) our choice of band-stop filter for removal of line noise. For inclusion of additional filters in the preprocessing pipeline, we have ensured the current design is flexible and extendable, and therefore, the pipeline can be easily extended with additional artifact-removing filters. For instance, we are currently implementing the Independent Component Analysis (ICA) algorithm for Android platform. For line interference and noise removal, although, we justify our use of band-stop filter, for future work, we are currently working on an optimized and Android-specific implementation of the CleanLine algorithm. Furthermore, we are running the aforementioned experiments on additional smartphone devices (including old models as well relevantly new models) to observe the behavior of proposed EEG preprocessing pipeline in terms of resource consumption. Finally, for further future work, we intend to apply the proposed smartphone app in the field to acquire and analyze EEG data in real-world experimental settings. This will be specifically done within the context of tinnitus research to gather EEG-related data of tinnitus patients.

Acknowledgment. This publication is a result of research supported by funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement number 722064 (European School for Interdisciplinary Tinnitus Research, ESIT) [29]. We would also like to acknowledge Brain Products GmbH (https://www.brainproducts.com/ Accessed: 15/06/2020) for their help and support while working with the LiveAmp 16 EEG amplifier.

References

1. Cleanline. https://www.nitrc.org/projects/cleanline. Accessed 04 June 2020
2. Abolfazli, S., Sanaei, Z., Gani, A.: Mobile cloud computing: a review on smartphone augmentation approaches. arXiv preprint arXiv:1205.0451 (2012)
3. Adjamian, P.: The application of electro-and magneto-encephalography in tinnitus research-methods and interpretations. Front. Neurol. **5**, 228 (2014)

4. Bigdely-Shamlo, N., Mullen, T., Kothe, C., Su, K.M., Robbins, K.A.: The prep pipeline: standardized preprocessing for large-scale EEG analysis. Front. Neuroinf. **9**, 16 (2015). https://doi.org/10.3389/fninf.2015.00016, https://www.frontiersin.org/article/10.3389/fninf.2015.00016
5. da Cruz, J.R., Chicherov, V., Herzog, M.H., Figueiredo, P.: An automatic preprocessing pipeline for EEG analysis (APP) based on robust statistics. Clin. Neurophysiol. **129**, 1427–1437 (2018)
6. Delorme, A., Makeig, S.: EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis. J. Neurosci. Methods **134**(1), 9–21 (2004)
7. Dohrmann, K., Weisz, N., Schlee, W., Hartmann, T., Elbert, T.: Neurofeedback for treating tinnitus. Progress Brain Res. **166**, 473–554 (2007)
8. Elgoyhen, A.B., Langguth, B., De Ridder, D., Vanneste, S.: Tinnitus: perspectives from human neuroimaging. Nature Rev. Neurosci. **16**(10), 632–642 (2015)
9. Esch, L., et al.: MNE: software for acquiring, processing, and visualizing MEG/EEG data. In: Magnetoencephalography: From Signals to Dynamic Cortical Networks, pp. 355–371 (2019)
10. Ganti, R.K., Ye, F., Lei, H.: Mobile crowdsensing: current state and future challenges. IEEE Comm. Mag. **49**(11), 32–39 (2011)
11. Güntensperger, D., Thüring, C., Meyer, M., Neff, P., Kleinjung, T.: Neurofeedback for tinnitus treatment-review and current concepts. Front. Aging Neurosci. **9**, 386 (2017)
12. Hassani, M., Karami, M.R.: Noise estimation in electroencephalogram signal by using Volterra series coefficients. J. Med. Signals Sens. **5**(3), 192 (2015)
13. Jastreboff, P.J.: Phantom auditory perception (Tinnitus): mechanisms of generation and perception. Neurosci. Res. **8**(4), 221–254 (1990)
14. Jastreboff, P.J., Hazell, J.W.: A neurophysiological approach to tinnitus: clinical implications. Br. J. Audiol. **27**(1), 7–17 (1993)
15. Kothe, C.A., Makeig, S.: BCILAB: a platform for brain-computer interface development. J. Neural Eng. **10**(5), 056014 (2013)
16. Kranczioch, C., Zich, C., Schierholz, I., Sterr, A.: Mobile EEG and its potential to promote the theory and application of imagery-based motor rehabilitation. Int. J. Psychophysiol. **91**(1), 10–15 (2014)
17. Lau-Zhu, A., Lau, M.P., McLoughlin, G.: Mobile EEG in research on neurodevelopmental disorders: opportunities and challenges. Develop. Cogn. Neurosci. **36**, 100635 (2019)
18. Levin, A.R., Méndez Leal, A.S., Gabard-Durnam, L.J., O'Leary, H.M.: BEAPP: the batch electroencephalography automated processing platform. Front. Neurosci. **12**, 513 (2018)
19. Lin, Y.P., Wang, Y., Jung, T.P.: Assessing the feasibility of online SSVEP decoding in human walking using a consumer EEG headset. J. Neuroeng. Rehabil. **11**(1), 119 (2014)
20. Mehdi, M.: Smart mobile crowdsensing for tinnitus research: student research abstract. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, pp. 1220–1223. ACM (2019)
21. Miyakoshi, M.: Makoto's preprocessing pipeline. Swartz Center for Computational Neuroscience (2018)
22. Mullen, T.: Cleanline EEGLAB plugin. Neuroimaging Informatics Toolsand Resources Clearinghouse (NITRC), San Diego (2012)

23. Oostenveld, R., Fries, P., Maris, E., Schoffelen, J.M.: FieldTrip: open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data. Comput. Intell. Neurosci. **2011**, 156869 (2011)
24. Pedroni, A., Bahreini, A., Langer, N.: Automagic: standardized preprocessing of big EEG data. Neuroimage **200**, 460–473 (2019)
25. Rajkumar, R., et al.: Comparison of EEG microstates with resting state fMRI and FDG-PET measures in the default mode network via simultaneously recorded trimodal (PET/MR/EEG) data. Hum. Brain Mapp. (2018)
26. Reiser, J.E., Wascher, E., Arnau, S.: Recording mobile EEG in an outdoor environment reveals cognitive-motor interference dependent on movement complexity. Sci. Rep. **9**(1), 1–14 (2019)
27. Renard, Y., et al.: Openvibe: an open-source software platform to design, test, and use brain-computer interfaces in real and virtual environments. Presence Teleoper. Virtual Environ. **19**(1), 35–53 (2010)
28. Schalk, G., McFarland, D.J., Hinterberger, T., Birbaumer, N., Wolpaw, J.R.: BCI 2000: a general-purpose brain-computer interface (BCI) system. IEEE Trans. Biomed. Eng. **51**(6), 1034–1043 (2004)
29. Schlee, W., et al.: Innovations in doctoral training and research on tinnitus: the European school on interdisciplinary tinnitus research (ESIT) perspective. Front. Aging Neurosci. **9**, 447 (2018)
30. Stopczynski, A., et al.: Smartphones as pocketable labs: visions for mobile brain imaging and neurofeedback. Int. J. Psychophysiol. **91**(1), 54–66 (2014)
31. Tadel, F., Baillet, S., Mosher, J.C., Pantazis, D., Leahy, R.M.: Brainstorm: a user-friendly application for MEG/EEG analysis. Comput. Intell. Neurosci. **2011**, 879716 (2011)
32. Xiong, H., Zhang, D., Wang, L., Chaouchi, H.: EMC^3: energy-efficient data transfer in mobile crowdsensing under full coverage constraint. IEEE Trans. Mobile Comp. **14**(7), 1355–1368 (2015)
33. Zhuang, Z., Kim, K.H., Singh, J.P.: Improving energy efficiency of location sensing on smartphones. In: Proceedings of the 8th International Conference on Mobile Systems, Applications and Services (MobiSys), pp. 315–330. ACM (2010)