ON THE INTERPLAY BETWEEN BRAIN-COMPUTER INTERFACES AND MACHINE LEARNING ALGORITHMS: A SYSTEMS PERSPECTIVE

A Thesis
Presented to
The Academic Faculty

By

Mohit Agarwal

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

May 2021

ON THE INTERPLAY BETWEEN BRAIN-COMPUTER INTERFACES AND MACHINE LEARNING ALGORITHMS: A SYSTEMS PERSPECTIVE

Approved by:

Dr. Raghupathy Sivakumar
Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Faramarz Fekri
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Chuanyi Ji
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Douglas M Blough
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Karthik Ramachandran
Scheller College of Business
Georgia Institute of Technology

Date Approved: December 8, 2020

ACKNOWLEDGEMENTS

Every accomplishment requires the effort of many people and this work remains true to that fact. I am highly indebted to my supervisor Prof. Raghupathy Sivakumar for his valuable guidance and support towards the completion of my dissertation work. Without his guidance and motivation, this dissertation would have never even seen daylight. I am honored to have worked under his excellent mentorship.

I would also like to thank Dr. Faramarz Fekri, Dr. Chuanyi Ji, Dr. Douglas M Blogh and Dr. Karthik Ramachandran for serving in my thesis committee and providing invaluable feedback on the work.

I thank all my labmates, Bhuvana Krishnaswamy, Chao-Fang Shih, Ekansh Gupta, Shruti Lall, Shyam Krishnan Venkateswaran, Uma Parthavi Morvapalle and Yubing Jian (in an alphabetically sorted order) for their helpful discussions and advice throughout this journey. I also thank my friends for being the wonderful people they are, and helping me out whenever I needed. Lastly and most importantly, I would like to thank my parents and my family for being there with me at every step of my life looking over my shoulder. Their unparalleled support and motivation has been the biggest driving force in my life and without doubt shall remain so.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Over the last century and a half, humans have provided input to computers through a range of technologies, including punch-cards, keyboards, mouse, stylus pens, and more recently through voice, and gestures. This enabled humans to interact with the computers and extend human capabilities across all knowledge domains, allowing them to make complex decisions underpinned by massive datasets and machine learning. Despite their obvious benefits, the action-based interfaces support only a handful of prescribed actions to perform, not only limiting the speed and effectiveness of communication but also degrading the communication experience. Furthermore, in most of the cases, action-based interactions fail to truly capture human intent due to the inherent channel loss in thought-to-action translation, limited linguistic capabilities, societal and psychological biases, etc.

At the same time, the seat of all human thought, and hence arguably the earliest manifestation of any intent to communicate, resides in the *brain*. Thus, it is natural to consider if it is indeed possible to tap directly into the brain to enable human to computer input. The notion of brain-based communication has existed for almost all of recorded history [1] and has been the subject of intense consideration for nearly a century and a half [2, 3]. EEG-based Brain-Computer Interfaces (BCIs) have emerged as a nascent modality to radically transform and redefine the way we communicate with computing systems. BCIs allow such information transfer by capturing brainwaves (synchronized neuronal firings) in the form of electrical potentials or EEG and decoding user intent with further analysis and processing. Since the first recording of human EEG (in 1929, Hans Berger [4]), for the last five decades, the primary application of EEG has been as a diagnostic tool to study conditions such as epilepsy and schizophrenia, and as an input modality for people with physical disabilities [5, 6]. While the development of portable *brain-computer interfaces* was instrumental in expanding the scope of applications EEG can be used for, consumer-grade EEG headsets are now available off-the-shelf and come in user-friendly form-factors

and are even fashionable [7]. The headsets have enabled the use of EEG as a casual input modality in niche applications such as gaming and wellness [8, 9]. Such devices have also opened up the opportunity for a study of the broader prospects of using EEG as a true first-class citizen amongst input modalities.

Simultaneously, machine learning, with its ability to automatically obtain deep insights and recognize unknown patterns in complex data sets, has seen remarkable successes in the past decade, in part by emulating how the brain performs certain computations. As we increase our understanding of the human brain, brain-computer interfaces can benefit from the power of machine learning, both as an underlying model of how the brain performs computations and as a tool for processing high-dimensional brain recordings. The technology (machine learning) has come full circle and is being applied back to understanding the brain and any electric residues of the brain activity over the scalp (EEG). Similarly, domains such as natural language processing, machine translation, and scene understanding remain beyond the scope of true machine learning algorithms and require human participation to be solved. This inherent inter-dependence and stimulating chemistry between EEG-based BCIs (brain) and computer algorithms (machine learning) is an attractive, emerging research area that must be studied and examined scientifically. Thus, investigating the interplay between brain-computer interfaces and machine learning systems through the lens of end-user usability forms the crux of this thesis dissertation. We study the interplay between EEG-based BCIs and ML algorithms at two different levels.

First, ML can be used as a powerful tool to learn and characterize the brain activity of an individual to build meaningful applications with it for day-to-day use cases. We characterize the battery life of BCI wearables and identify the issue of the low-battery life of commercially available BCI wearables. We performed experimental power analysis to gain insight into what micro components of BCI wearables can be used as a control knob to operate BCI wearables in low-power mode. We studied the practical benefits of using eye-blinks as a command modality and proposed *BLINK* algorithm to detect eye-blinks through

EEG in a completely automatic and unsupervised manner. We proposed *Trance*, a wake-up command detection system to increase the battery life by 2.7x with real-time detection of BCI wearable.

Second, EEG-based human feedback can fundamentally help ML algorithms either as an alternative input or implicit feedback. Providing user-thoughts as the direct input to the ML algorithms could be beneficial in designing advanced personalization systems or high precision information retrieval tools. We consider a paradigm where user personalization models, specifically, preferences for online merchandise, are created based on the user's thoughts alone. We propose *Cerebro*, capable of ranking consumer products according to the user preferences by relying solely on the user's brainwaves. For the learning algorithms that require significant human involvement, EEG can provide feedback directly without putting any burden on the user. We study how intrinsic reactions captured through EEG (in the form of error-potentials) can accelerate the learning of RL agents, and develop an end-to-end system to enable such a paradigm.

CHAPTER 1
INTRODUCTION

The brain is the seat of human intelligence, cognition, and behavior [10]. Hence, for most of known history, humans have conceptualized, fantasized, and explored the notion of communication directly through thoughts in the brain [11]. With the discovery of electroencephalography (EEG) in 1929, obtaining a simple window into the functioning of the brain became a reality [4]. At a high level, any brain activity occurs through the synchronized electrical firing of billions of brain cells (neurons) communicating with each other. Such activity can be detected externally through appropriate sensors on the scalp over the brain, enabling the direct transfer of information from the human brain to a computer, also known as brain-computer interfaces (BCIs). BCIs arguably provide a better modality of communication for human-computer interface applications because they are *non-intrusive compared to other input modalities, enable the capture of passive user intent, allow for shortened intent to action pathway latency, and provide high degrees of privacy.*

Today, computer algorithms use traditional human-computer interfaces (e.g., keyboard, mouse, gestures, etc.), to interact with and extend human capabilities across all knowledge domains, allowing them to make complex decisions underpinned by massive datasets and machine learning. For example, IBM Watson relies on an enormous database of cases to recommend the best treatment strategy for a cancer patient to oncologists [12]. This growing interaction between humans and computers in a symbiotic fashion is delivering intelligence, productivity, and enhanced communication experience at an unprecedented scale. Machine learning, with its ability to automatically obtain deep insights and recognize unknown patterns in complex data sets, has seen remarkable successes in the past decade, in part by emulating how the brain performs certain computations. It consists of neural networks, similar to the brain's network of neurons, giving it the ability to distinguish an image of a dog from one of a cat, to use a camera feed to spot vehicles and pedestrians

while navigating a self-driving vehicle and to understand and communicate in the natural language. As we increase our understanding of the human brain, brain-computer interfaces can benefit from the power of machine learning, both as an underlying model of how the brain performs computations and as a tool for processing high-dimensional brain recordings. The technology (machine learning) has come full circle and is being applied back to understanding the brain and any electric residues of the brain activity over the scalp (EEG). Similarly, domains such as natural language processing, machine translation, and scene understanding remain beyond the scope of true machine learning algorithms and require human participation to be solved.

The inherent inter-dependence and stimulating chemistry between EEG-based BCIs (brain) and computer algorithms (machine learning) is an attractive, emerging research area that must be studied and examined scientifically. Thus, investigating the interplay between brain-computer interfaces and machine learning systems through the lens of end-user usability forms the crux of this thesis dissertation. Specifically, we explore how these two highly powerful, yet complementary entities can benefit each other, and propose systems and algorithms for achieving the same. In this context, we provide our research contributions in two inter-related aspects by, (i) applying machine learning to solve challenges with EEG-based BCIs, (ii) enabling human-assisted ML with EEG-based human input.

The global BCI market was valued at $1.15 billion in 2018, and it is expected to reach $2.67 billion by 2026[1]. However, the current consumer market presents a few consumer-grade brain wearables. These are hardly being used outside the clinical or research settings. Despite their established promise, the adoption of the technology has made very slow strides and is strictly limited to niche applications. The use and development of brain-computer interfaces, historically, has always been driven by the needs of users with motor-disability functions. It has not been thoroughly subjected to practical communication barri-

[1] https://www.globenewswire.com/news-release/2019/07/22/1885929/0/en/Brain-Computer-Interface-Market-To-Reach-USD-2-67-Billion-By-2026-Reports-And-Data.html

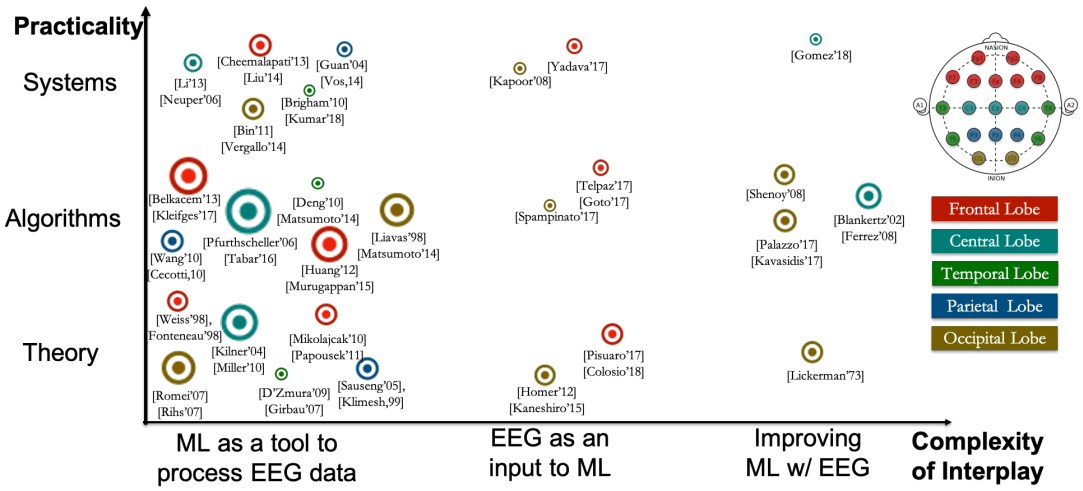


Figure 1.1: Landscape of BCI and ML research

ers (e.g., large form-factor, user discomfort, low bit-rate, and accuracy, etc.). These devices were not designed to fit into the lifestyle of the mass consumer (functionality, comfort, and cultural aspects), and their applications were limited to off-site laboratory tests.

1.1 BCI Research Landscape

We present a high-level research landscape of research at the intersection of EEG-based BCIs and ML algorithms. We present this landscape along with the complexity (or sophistication) of the interplay between BCI and ML algorithms (on the x-axis). On the y-axis, we have the practicality of the systems in three major categories, theory, algorithms, and systems. Further, we provide the color-coding of the research works in terms of the major lobes (and hence, functionalities) of the human brain. The size of the dots reflects the relative amount of work done in a particular area.

At the left-bottom corner, there is an abundance of research work performed in understanding the basic functionalities of the human brain, and correlating such aspects with the observed variations in the brainwaves. Cognitive and working memory performance were found to be correlated with alpha and theta oscillations [13, 14]. Several works studied the process and comprehension of natural language [15, 16] and sensing of imagined speech [17, 18]. Excitability and spatial attention in the visual cortex was found to be correlated

with alpha band activity in EEG [19, 20]. Further, cortical activity was investigated during imagined visuomotor tasks [21], motor imagery based online feedback [22], and emotional intelligence [23, 24]. Various ML based classification algorithms were proposed in the literature for motor imagery tasks [25, 26], eye-blink [27, 28], emotions [29, 30], imagined speech [31, 32], visually-evoked potentials [33], P300 potentials [34, 35], etc. Based on the theory and algorithms, systems were researched and designed for control of wheelchair [36], spelling devices [37, 38, 39], subject identification [40], etc.

As we move along the complexity of the interplay, the density of the research work reduces considerably. Evidences of value-based decisions [41], subjective preferences [42] and choice-induced preferential changes [43] were established with the neural correlates. [44] analyzed the representational similarity of the object processing dynamics. [45] developed an algorithm to predict future consumer choices by relying on EEG data. An automated object classification system driven by human brain signals was proposed [46].

Brain2Image [47] is capable of generating images using visually-evoked EEG potentials. An end-to-end system was developed to correct the mistakes of a robot in real-time using primary and secondary error-potentials captured through EEG [48].

1.2 Research Contributions

Our research contributions primarily lie in the domain of system and algorithms, investing the interplay between EEG-based BCIs and ML algorithms through the lens of end-user usability. We provide out research contributions at two levels, as below,

1. Using ML techniques to solve challenges in EEG processing : One of the most debilitating aspects of EEG is its vulnerability to distortions caused by other interfering electrical fields, especially eye-blinks, leading to confused or possibly false EEG interpretations. Hence, the detection and removal of eye blink components are imperative in any EEG-based intent decoding and analysis. The currently available solutions suffer from one of the limiting requirement - (i) a partly manual inspection for thresholds or template

selection, (ii) a user training phase, (iii) a high number of EEG channels, and (iv) Electrooculography (EOG) channels requiring additional electrodes above and below the eyes. We contribute a fully automatic and unsupervised (i.e., *without requiring any training from the user*) blink detection algorithm, *BLINK*, capable of identifying accurate timestamps of eye blinks in a single-channel EEG data. Further, building upon the eye-blinks, and their detection through *BLINK*, we propose a wake-up command design and detection for BCIs, and explore how battery life can be made to last for approximately a day (2.7x, 10.14 hours). With our preliminary motivational study with currently available commercial BCIs, we found that the wearable BCI headsets are always-on and are thus power-hungry, requiring users to charge headsets multiple times a day. The key challenge that we address is enabling the cap to operate in a near-sleep mode while still reliably detecting and interpreting a wake-up command from the user. Our core contribution is *Trance*, a user-friendly and robust wake-up command for BCI that is computationally lightweight and hence can be supported by off-the-shelf BCI caps.

2. Enabling human-assisted machine learning with EEG-based human input: Machine learning algorithms developed for recommendation engines on e-commerce platforms (e.g., Amazon.com), digital media (e.g., Netflix), and advertising platforms (e.g., Google AdSense) rely on user models for personalization. The current paradigm constructs and adapts the user models based on user-actions whether explicit actions (e.g., ratings, reviews) or implicit actions (e.g., browsing history, click-through rate) are used. In this work, we argue that a machine learning algorithm that also takes into account user-thoughts is likely to be significantly more informative and accurate than one that relies on user-actions alone. The specific goal in this context is to determine the preference ranking for a set of objects by relying entirely on the brain activity of a user recorded through a wearable EEG headset. We present a machine learning algorithm, *Cerebro*, which can learn the specific nuances of the user's brainwaves and rank the objects accurately based on preferences. We measure the accuracy of the algorithm in terms of the Normalized Discounted Cumulative

Gain (NDCG) score, showing that it performs with an attractive score of 0.92 when trained on 7 objects, and evaluated on 3 objects for the 14 users.

Despite the tremendous advancements in machine learning, there are still several frontiers that remain unsolved. Several AI-complete problems rely on human participation, either during the training phase or while using the algorithm in live situations. In this work, we explore an interesting solution paradigm that will allow humans to assist machine learning algorithms without being over-burdened. This model benefits from the natural rich activity of a powerful sensor (the human brain); but at the same time, it does not burden the human if an activity is intrinsic. This paradigm is inspired by a high-level error-processing system in humans that generates error-related potential (ErrP), a negative deflection in the ongoing EEG. We develop three reasonably complex 2D discrete navigational games to experimentally evaluate the overall performance of the proposed work. Major contributions of our work are as follows: i) we propose and experimentally validate the generalizability of error-potentials, where the error-potentials can be learned for one game, and transferred to other unseen games, (ii) we propose a novel RL framework for integrating implicit human feedback via error-potentials with RL agent, improving the labeling efficiency and robustness to human mistakes[2], (iii) we propose an algorithm for reliable detection of error-potentials through user brainwaves, and (iv) compared to prior works, we scale the application of error-potentials to reasonably complex environments and demonstrate the significance of our approach for accelerated learning through real user experiments. We show that with the proposed algorithm modifications, error-potentials can be decoded with 84.4% accuracy (11.05% improvement) and can achieve acceleration upto 3.38x while making 75.56% fewer queries.

[2]The RL framework, i.e., how error-potential based human feedback is integrated with RL algorithm, is the contribution from our collaborators, Duo Xu, and Dr. Faramarz Fekri. Their contributions are explained here in a very succinct manner for completion purposes. We thank our collaborators for their outstanding work and for providing us the permission to present their work in this thesis.

1.3 Thesis Statement

Machine Learning (ML) algorithms and Brain-Computer Interfaces (BCIs) can be used synergistically, with ML enabling BCI and BCI assisting ML, and the interplays are demonstrable using real-life applications.

1.4 Thesis Organization

The thesis dissertation is organized as follows. Chapter 2 presents the relevant literature survey. In chapters 3 and 4, we explain our contributions in using machine learning for solving usability and wide-adoption challenges in EEG-based BCIs, i.e., *BLINK* and *Trance*, respectively. In chapter 5, we introduce how machine learning algorithms can be informed with human preferences using EEG as an alternative input and propose *Cerebro* to rank objects based on the user's brainwaves. In chapter 6, we explain the system and research allowing EEG-based BCIs to assist Reinforcement Learning (RL) algorithms. Finally, we discuss additional challenges and further research directions in chapter 7.

CHAPTER 2

LITERATURE SURVEY

Inspired from the early works of Richard Caton [49] and Vladimir Neminsky, [50] Hans Berger published the first-ever work on the human EEG and the presence of alpha waves in 1929 [4]. The first BCI dates back to 1973, developed by Jacques Vidal to control the cursor movements. He used the expression for his research projects at UCLA [51, 52], funded by NSF and contracted by DARPA, which marked the beginning of research in BCI for communication and control. In this chapter, we provide a summary of the relevant research and compare them with the proposed contributions of this work.

2.1 BCI Systems

Historically, BCI systems have always been considered for biomedical applications to develop assistive devices. The developed BCI technology was slow (in terms of data rate) and hence, targeted locked-in individuals or physically-challenged users suffering from various neuromuscular disorders. However, the recent breakthroughs in the field of sensor design (hardware), signal processing and machine-learning (software) along with the increased understanding of brain functionality (core-neuroscience), allowed to widely expand the focus of BCI systems to include improved communication and Human-Computer Interface

Figure 2.1: Monkey feeding itself using invasive-BCI
(Image taken from [53])

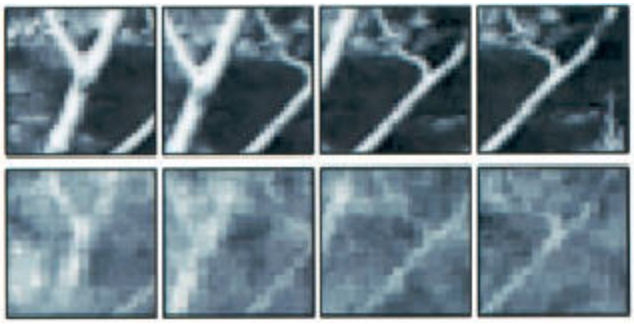


Figure 2.2: Restoration of images seen by cats
First row displays actual image, second row displays restored image (taken from [54])

(HCI) experience for healthy users. Today, with the improved hardware capability and usability, BCI systems are becoming commercially relevant to expand beyond the communication needs (and interfacing applications) to include numerous innovative aspects including health monitoring, security, education etc.

Technically, the terminology BCI systems encompasses the galaxy of systems that have the capability to tap into a primate's brain through magnetic (fMRI, MEG) or electric (EEG) sensors placed inside (invasive) or outside (non-invasive) of the brain. For the purpose of this proposal, we restrict BCI systems to only non-invasive EEG based systems. Such a typical BCI system consists of three main components, (i) an electrode sensor array placed on the scalp (ii) a hardware platform to digitize crude brainwaves, and (iii) an algorithmic processing platform to interpret brain waves. A scalp electrode array is the cornerstone of the BCI system, providing a conductive medium for brainwaves to reach the hardware interface. Typically, the electrode array (a set of electrodes) is positioned over a cap or in the form of a wearable headset. The hardware platform digitizes and amplifies the crude brainwaves captured from the scalp electrodes. The digitized signals are shipped to the software counterpart where a series of code-based processing is performed to map high-dimensional (read, massive) complex EEG signals to trivial application-based outputs.

2.1.1 Invasive BCI

Various researchers across the world started to develop and plant electrodes inside the grey matter of the brain in living beings, termed as invasive BCI. Experimental studies performed on monkeys and rats revealed the possibility of voluntary control of external devices using neural signals [55, 56, 57]. Rhesus monkeys were trained to use BCI to track visual targets or feed themselves (Figure 2.1) using robotic arms [53]. Researchers from the University of California Berkeley were able to reproduce images seen by cats by decoding neuronal firing patterns in the brain's sensory input area (Figure 2.2) associated with retina [54].

Human BCI implantation for medical and control purposes increased significantly after the 1970s, restoring locomotion, vision impairment, neuromuscular and mental disorders, marking several seminal discoveries in the field [58, 59, 60, 61]. Phillip Kennedy was credited for the first BCI implant in humans. His patient was Johnny Ray who was suffering from 'locked-in syndrome' at the time of implantation, learned to control a computer cursor, and died later in 2002 [62]. Matt Nagle, suffering from Tetraplegia, was the first person able to control an artificial hand using BCI developed by Cyberkinetics [63].

2.1.2 Non-invasive BCI

While invasive BCI has its merits over non-invasive techniques, in terms of signal quality and noise sensitivity, non-invasive techniques gained attention for healthy users as it does not require surgical procedures and in-head implantations which often pose risks to human health. The main non-invasive BCI technologies are electroencephalography (EEG), magnetoencephalography (MEG), and functional magnetic resonance imaging (fMRI). Neuronal activity inside the brain generates electrical activity that is captured by EEG, while MEG captures produced magnetic fields (due to electric currents). fMRI on the other hand measures changes in blood flow inside the brain cell to determine brain activity. EEG is one of the most widely used non-invasive technology due to its temporal resolution, safe, easy,

and inexpensive procedure. Today, in spite of the presence of high-resolution MRI, EEG is popular in medical use for diagnosing epileptic seizures, sleep disorders, brain death, etc., [64, 65]. EEG applications for healthy users mainly fall into categories, namely, enhanced communication and remote control, cognitive performance improvements, gaming, neuromarketing, and brain computing [66, 67, 68, 69]. The prominent electrophysiological signals used to design present BCI systems are Visually Evoked Potentials (VEPs), Event-Related Potentials (ERPs), Slow Cortical Potentials (SCPs), and sensorimotor rhythms.

Visually Evoked Potentials (VEPs): Stimulating a subject's central or peripheral visual field evokes large potentials in brain signals, dominant in the occipital scalp area. It has been established that occipital brain frequency resonance with the frequency of visual stimuli, oscillating in a sinusoidal pattern. These are further categorized into transient VEP (tVEP) and steady-state VEP (SSVEP) based on stimulus rates. Vidal developed a VEP-based BCI which could move a cursor on a monitor screen by determining the eye gaze direction of the user [70, 71]. Brain Response Interface (BRI) developed by Sutter (1992) presented an 8x8 grid of symbols and achieved a rate of 10-12 words/min with high accuracy [72]. [73] designed a self-regulated BCI and achieved an accuracy rate of 92% with an avg. selection time of 2.1s.

Slow Cortical Potentials (SCPs): As evident from the name, slow cortical potentials are slow oscillations that could last upto 10s. SCPs are typically associated with cortical activation [74, 75, 76], which can be learned to control with training procedures. Various Thought Translation Devices (TTD) were demonstrated based on SCPs, extensively targeted for providing communication abilities to locked-in patients [77]. Similarly, SCP based BCI, Language Support Program (LSP) can write 2-36 words/hr with accuracy ranging from 65 to 90% [78].

Event-Related Potentials (ERPs): ERPs are behavioral responses of the brain to specific events or infrequent (or significant) stimuli infused with regular stimuli in auditory, visual, or sensory format. P300, a positive deflection after 300ms of stimuli, is predominantly used

in several modern BCI designs. One of the famous P300 based BCI is the P300 speller, initially developed by Farewell and Donchin in 1988 with an information rate of 5 letters per minute and improved further in upcoming years[79, 80]. A typical 'P300 speller' presents a 6x6 matrix of symbols flashing rows and columns with distinct frequency, requiring users to pay attention to a particular symbol. N170 presents a negative peak after 170ms correlated with facial visual stimuli, helpful in distinguishing cases of faces vs non-faces [81]. Similarly, other ERPs, namely N400, N300, P600, etc are associated with semantic congruity and language processing [82, 83].

Sensorimotor Rhythms: Sensorimotor rhythms, also known as mu waves are EEG activities occurring over sensory and motor cortical areas of the brain, in between frequency range of 8-12 Hz. They occur with actual or imagined body part movements and are distinct in terms of spatial localization over the primary motor and sensory cortex of the brain, mapped directly to the motor and sensory body parts [84]. The Wadsworth BCI is based on the same signals, which require users to imagine limb movements to control a cursor on a 2-D screen. The system achieving an information bit rate of 20-25 bits/s [85] requires elongated training and is hectic in terms of its use operations. Mu-waves based BCIs are particularly favored as they don't present strict requirements to external stimuli. [86] explores a similar problem to facilitate communication-based on thoughts itself.

Emerging areas in non-invasive BCIs: One of the emerging areas in BCI is human-aided computing. [87] builds a classification system exploiting cognitive processing in human brains. Performance of current state-of-the-art computer vision algorithms is shown to improve when combined with implicit human processing and EEG [88]. [67] focuses on cognitive performance improvement by monitoring brain activity during daily mental and physical activities. Blending of BCI technology with virtual reality systems is slowly transforming the interactive education and entertainment world [89].

With increased knowledge of frequencies and patterns exhibited by brainwaves, and the advent of novel technologies is motivating researchers and entrepreneurs to jump into the

consumer sector of BCI. Mindflex by Mattel, Mindwave by NeuroSky, and Star Wars Force Trainer are few inexpensive EEG-based consumer BCI products in the entertainment sector [90, 91]. Mind Solutions Inc., launched NeuroSync [92] a brainwave optimization device designed for relaxation, self-regulation, and meditation. Emotiv's EPOC/EPOC+/Insight and NeuroElectrics's Enobio [93, 94] develops EEG based BCI headsets in wearable design using dry sensor technology for communication, control, and gaming purposes.

2.1.3 Consumer grade BCI systems

Competition in the corporate sector today presents a multitude of options for research- and consumer-grade BCI systems available in the market. These systems are either available in bundled packages as wearable (with all three components), or present top-notch solutions for one of the components in the BCI system. Here, in this subsection, we describe a few leading products available in the market for the same.

OpenBCI [Hardware Interface]

Joel Murphy and Conor Russomanno developed OpenBCI[1] (Open Source Brain-Computer Interface) which is an open-source, low-cost, programmable interface to access raw EEG signals. The interface has the capability to connect with up to 16 electrodes at a time, amplifying and digitizing the signals at 250Hz. It is built around the Atmel ATmega microcontroller, which can be re-programmed on the board. The heart of the OpenBCI board is ADS1299[2], designed and manufactured by Texas Instruments. It is a multi-channel, low-noise, 24-bit Analog-to-Digital Converter (ADC) specifically designed for EEG and similar biopotential measurements. It can also be used for measuring muscular and heart activity i.e. electromyography (EMG) and Electrocardiography (ECG) respectively. The most recent version of OpenBCI i.e., v3, comes with RFduino and USB dongle which allows digitized EEG signals to transfer to PCs, laptops, smartphones, or any Bluetooth compatible device in a wireless manner. The installed RFduino is equipped with the latest

[1] https://openbci.com/

[2] https://www.ti.com/product/ADS1299

Bluetooth Low Energy (BLE) technology, which supports similar data rates as the older version with reduced power consumption, making it last longer. OpenBCI board is also armed with an SD card slot to store signal data in a memory card for situations where instantaneous connectivity is not possible.

A lot of such hardware modules are available in the current market but either they are very expensive, or perform poorly or provide restricted access to system design and raw EEG data. It is very crucial for the research community to have all of the above features bundled in a single piece of hardware. What makes OpenBCI unique and suitable for our purpose, is its transparent design with full control over raw EEG signals as well as access to hardware architectural design and underlying algorithms for translating EEG signals to meaningful data, which can be expanded or modified further to suit our needs. OpenBCI comes with Brainwave Visualizer written in Java, C++, and Python. It can be used to simultaneously visualize time-domain EEG signals, their frequency spectrum, and spatial power localization.

Emotiv EPOC+ [Wearable]

Emotiv EPOC+ [3] is a research-oriented wearable EEG headset manufactured by Emotiv, Inc. and launched in 2013. The headset consists of saline-based wet sensors, providing a smooth and non-sticky interface to capture the raw signals from 14 different positions across the scalp. The headset has the capability to connect to a desktop or smartphone through Bluetooth technology. The bundle EPOC+ package also includes the SDKs and applications for various platforms (Windows/macOS, Android/iOS) along with the software packages. It provides access to the dense array, high quality, raw EEG data for research purposes using the subscription-based software, EmotivPRO. The headset is lightweight and includes a rechargeable Li-ion battery which is claimed to last 12 hrs of continuous use.

[3] https://www.emotiv.com/epoc/

Muse [Wearable]

Muse [4] was launched in 2014 by InterAxon Inc. and is marketed as a brain-sensing headband for the primary purpose of meditation and self-regulation oriented applications. Muse consists of 4 dry scalp-sensors that can measure the amount of brain activity and a relaxed state of mind. Muse comes with a smartphone application to measure and self-assess the cognitive state of the users. Developers around the globe have built successful applications to read real-time raw EEG data recorded by Muse to control smartphones, play games, etc.

BIOPAC CAP 100C [Electrode-sensor]

BIOPAC Cap 100C [5] is a sensor array with 19 touch-proof electrodes to record EEG with less application time and increased user-comfort. The cap fabric (made of Lycra) is stretchable and extremely comfortable. Electrode locations are fixed according to the International 10/20 system, to minimize the placement errors. An electrode gel needs to be injected at the electrode positions to establish the conductive medium. The cap ships with a connector allowing it to attach to any hardware interface system.

g.BCISys from g.tec [research grade]

g.BCISys[6] is a high-end research grade standard solution used across many BCI research labs across the world for bio-signal acquisition, amplification, and processing. It has the capability to sample 16 channels simultaneously at 40kHz with the bit-resolution of 24-bits. The amplifier is FDA and CE certified. The high sampling rate and low noise floor ($\leq$ 0.35 uV) allow detection of evoked potentials (EPs) which are typically not possible with conventional amplifiers. The system has impedance check subroutines, which beeps if the signal quality of any particular electrode drops below the minimum required threshold level, thus, ensuring high data quality at all times.

[4] https://choosemuse.com/
[5] https://www.biopac.com/product/eeg-caps-for-cap100c/
[6] https://www.gtec.at/product/bcisystem/

EEGLAB [software module]

EEGLAB [95] is an open-source interactive MATLAB toolbox for processing and analysis of electrophysiological data released by Swartz Center for Computational Neuroscience [7] at University of California, San Diego. The toolbox provides an interactive and easy-to-use graphical interface to load, process, and visualize various aspects of the EEG data. EEGLAB is compatible with various bio-signal data formats (e.g. BioSIG, BDF, GDF etc). The toolbox has several processing and visualization algorithms required to analyze event-related EEG and other bio-signals, including time/frequency analysis, Independent Component Analysis (ICA), artifact rejection, forward/inverse head/source modeling, event and channel location handling, etc. The toolbox is available across platforms spanning Linux, Windows, and macOS.

OpenViBE [software module]

The software counterpart of our research project is OpenViBE developed by Inria, INSERM, and Orange Labs [96]. It is free software distributed under an open-source license, for designing, testing, and using BCIs. It can acquire, filter, process, classify and display EEG data in a real-time environment. Its open-ended design and availability of numerous data acquisition drives allow it to directly interact with any BCI system including OpenBCI.

OpenViBE is written in C++, compatible with both Windows and Linux environments. It comprises numerous software modules devoted to data acquisition, algorithms for signal filtering, digital signal processing, machine learning, pattern recognition, and data visualization, which can be interconnected to design a BCI software paradigm. Software users without any programming experience can design a successful BCI system, using its graphical user interface without even writing a single piece of code. It has abstract and categorized representations of all software algorithms. Researchers and programmers can even modify the code or develop such software blocks on their own to add more functionality to their

[7] `https://sccn.ucsd.edu/`

BCI design. It can interact with various high-end Virtual Reality (VR) applications, which makes this platform a top choice for neuro-game developers.

Pre-configured scenarios for multiple BCI paradigms are present in OpenViBE including motor imagery design, P300 speller, etc. to get a head start in designing BCI systems.

2.2 EEG-based Eye-blink Detection Algorithms

Several related works lie at the intersection of EEG and eye-blinks, which can be broadly classified into two categories: (i) removing eye-blink artifacts from the EEG signal, and (ii) detecting the time instants of eye-blinks in EEG. From a technical perspective, both categories are quite different from each other. The former removes the eye-blink components from EEG resulting in pure cerebral data, however, is unable to locate the time instants of eye-blinks. The latter locates the time instants but is incapable of removing the distortion without losing the cerebral data within the eye-blink duration. Several hybrid approaches have been proposed in the literature first to identify the eye-blinks and removing the related component to clean the signal [97, 98].

2.2.1 Blink component removal methods

Multiple strategies are proposed in the literature to purify the EEG waveform using Blind Source Separation (BSS) based methods. These methods [99, 100, 101, 102, 103] vivisect EEG waveform into additive subcomponents using BSS algorithms like Independent Component Analysis (ICA) and remove the non-cerebral (mostly eye-blink) component from the EEG using template matching. The templates are created with labeled eye-blink examples which are proved to be consistent across users. These methods perform very well but require maintaining a large database of templates, and sampling from a large number of electrodes to find the multiple subcomponents. [104] is one such semi-automatic process requiring the manual labeling and selection of a template. Some of these works even require putting extra electrodes over and above the eye, also known as Electrooculography (EOG) [101]. EyeCatch [105] uses a similar strategy to detect eye-blinks specifically. It

analyses and compares the IC scalp maps with the half-million scalp maps present in their database. ICA-based approaches are advantageous in circumventing the limitations of conventional artifact detection methods, however, they can be only used in dense EEG systems due to their strict requirements of a high number of EEG channels.

[106] presented a new identification procedure based on an efficient combination of independent component analysis (ICA), mutual information, and wavelet analysis for fully automatic ocular artifact suppression. The results on 3105 4-s EEG epochs indicate that the artifact components can be identified with an accuracy of 97.8%, a sensitivity of 96.9%, and specificity of 98.6%.

Various non-ICA based approaches were proposed in the literature to tackle the limitations of ICA based approaches. [107] combined Discrete Wavelet Transformation (DWT) and Adaptive Predictor Filter (APF), [108] combined EMD and CCA, [109] used adaptive filtering, [110] used autoregressive moving average exogenous (ARMAX) model with extended least square (ELS) algorithm, [111] used RBF (with adaptive optimization) to remove the ocular artifacts from the EEG signal. Methods proposed in [112, 113] are capable of removing such artifacts using only a single EEG channel through combining Singular Value Decomposition (SVD) with Singular Spectrum Analysis (SSA), and algebraic and DWT methods respectively. The recently proposed approach [114] combines morphological component analysis (MCA) and k-SVD to achieve the same. [115] uses a model based on the ballistic physiological components of the eye blink and achieves a success rate of over 90% in terms of recovering the variance of the original EEG.

The method proposed in [112] is capable of removing such artifacts using only a single EEG channel through Singular Value Decomposition (SVD) and Singular Spectrum Analysis (SSA). [113] uses algebraic and DWT based methods to remove such artifacts using only single-channel EEG data.

2.2.2 Blink identification methods

A very trivial approach to detect eye-blink timestamps is to continuously monitor the EEG signal and detect eye-blink if the amplitude crosses a preset threshold value. Improved approaches in the literature extract relevant features to apply a threshold. In [116], various statistic based features were calculated for data artifacts in five aspects of the EEG data: channels, epochs, ICs, single-channel single-epochs, and aggregated data (i.e., across subjects). A threshold of ± 3 was used for the Z-score for each feature to detect the blink artifact. [116] was shown to perform with a score of 94.47 and 98.96 for sensitivity and specificity, respectively on simulated data over 128-channels. The performance of [116] drops significantly with a reduced number of electrodes (i.e., 32). [117] employs the use of extreme statistics and used p-value as the threshold parameter to detect the blink artifacts on 29-channel EEG data. An automatic threshold of $\mu + 2\sigma$ is used along with channel correlation (in Fp1 and Fp2) electrodes in [118]. [119] proposed the use of multi-window summation of derivatives approach and compared against the correlation, Dynamic Time Warping (DTW) and Root Mean Square Error (RMSE) based approaches. A similar threshold-based approach was used in [120] along with DTW. [121] applies a threshold-based peak detection technique for activating the home lighting system. An intelligent approach over simple amplitude thresholding is to extract relevant features from the EEG signals and perform binary classification by comparing it with a threshold. Such threshold-based techniques were also used in [117] over the frequency spectrum. Power Spectrum Density (PSD) of a moving window was compared to a threshold to detect eye-blink artifacts. The performance of such methods suffer due to a high variance in eye-blink duration, and blink peak not falling in the middle of the window. Threshold-based approaches are highly sensitive to the chosen features and preset threshold, which could vary highly across devices and subjects.

Fingerprint or template matching based methods are widely used in the field of pattern recognition. In these approaches, an eye-blink template (or fingerprint) is first obtained and then matched with the continuous EEG data using a moving window. If the similarity

measure crosses a preset threshold value, an eye-blink signal is detected in the particular window. These methods are highly sensitive to the chosen template and the similarity metric. [122] applied Dynamic Positional Warping (DPW), a variant of DTW, and demonstrated the accuracy improvements over DTW, RMSE, and correlation as the similarity metric. The templates are typically chosen either through manual inspection or generated with an algorithm. [122] selected five templates from the ground truth dataset, and hence is not fully unsupervised.

Supervised-learning based methods design a specific kind of neural network architecture (or deep architecture) for learning the distinctive and similar patterns based on the training data [97, 123]. [123] uses Support Vector Machines (SVMs) for the identification of blink artifacts with a moving window of 450ms. [124] uses segmentation of a 1-second window and applies the RBF network on three extracted features achieving an accuracy of 75.3%. Such techniques demand user-training and are heavy in computation (for training) and memory (weight storage).

Other algorithms that work on purely statistical techniques do not estimate the blink positions but instead count them [125, 126] or are highly sensitive to the input parameters. [126] does not explicitly detect eye-blinks but any spiked artifacts. This can result in high false positives as a result of eye and head movements. Sensitivity to the input parameters defeats the universality point. [98] proposed a complicated approach of combining a high-speed eye tracker to timestamp blinks and further removed artifacts caused by eye-blinks and movements. [106] proposed a novel combination of ICA with mutual information and wavelet analysis to achieve 97.8% accuracy using 6 EEG and 2 EOG electrodes. [127] detects blink artifacts with 90% specificity and 65% sensitivity using an extended Kalman filter. [128] performs DTW score clustering during wearable EEG-based cognitive workload assessment tests to achieve an accuracy of 96.42%. Despite the attractive performance rates, the proposed method is not suitable due to the requirements of user training and 7-EEG channels. [28] relies purely on statistical techniques but requires the EEG signal for

an extended period of time (offline), to extract the blink profile. Regression-based methods require measuring EOG electrodes to correctly estimate the regression coefficients [129, 130, 131]. This again puts extra hardware requirements on the available EEG architectures in the market and is clearly not suitable for our case; hence, we skip the discussion of such approaches. Thus, there does not exist any eye-blink detection algorithm (through EEG) that fits the requirements of universality, no supervised-training, no manual involvement, small form-factor, and near-perfect detection accuracy. In this context, we later present in this thesis, a novel solution and compare it against a specific related work, BLINKER [28].

2.3 Eye-blinks as an Input Modality

Eye blinks are widely used as a communication modality in smartphone and VR applications for home automation, gaming, snapping photos, etc [132, 133, 134, 135, 136]. The primary reason behind this is their naturality and ease of use. Various eye-based systems, e.g., eye-gaze, wink, blinks, eye-movement tracking, are presented in the literature as an interaction modality between humans and machines [137, 138, 139, 140, 141]. Tag et al. [142] proposed a real-time system adapting video settings as per the viewer state. The viewer state is described as the average eye-blink frequency measured through electro-oculography. Pike et al. [143] used eye-blink, levels of attention, and meditation (recorded through EEG), to influence the adaptive media. Huang et al. [144] presented PACE, to collect user-interaction data unobtrusively by relying on the eye and facial analysis of webcam data. In [145], Chatterjee et al. argued that combining eye-gaze with gestures can outperform the individual, and in general, approach the gold-standard performance of input systems (e.g., mouse, trackpad, etc.). "Blink Link" [140] was designed by Grauman et al. leveraging a series of eye-blinks as an alternative communication tool for users with severe disabilities through computer vision processing. In our work, we focus on using eye-blink detection through EEG-based BCI wearables, and only to deliver a wake-up command.

2.4 Detection of EEG-based Implicit User Preferences

The detection and analysis of consumer preferences through EEG based neuro-biological changes has been studied thoroughly. [146, 147, 148] studied the extensions of brands to different product categories. Studies in [146] revealed that N270 is directly associated with the conflict in the brand category and the extended category, thus, can be used as a reference in brand extension attempts. [147] and [148] shows a similar association of P300 potentials and N400 potentials for mental categorization in brand extensions. [149] uses K-nearest neighbors and probabilistic neural nets on Alpha wave features to recognize the most preferred automotive brand with 95% accuracy. Authors in [150] show a positive correlation between the passive viewing of luxury (branded) goods with the Late Positive Potentials (LPPs) in EEG, in the presence of another person. [151] uses the LPPs to relate the olfaction and emotions, and provide insights on the emotional reactions of the consumers to the ambient scents. [152] explored the positive relationship of LPPs with the herding tendency of consumers in the context of online reviews for book purchases. In the same context of recommendations, [153] validates similar herding behavior through P300 waveforms.

In a consumer shopping task, [154] explored the ERP measures and the role of math anxiety in consumers for discounted and promotional products. The correlation of different EEG frequency bands with the subject's internal decision of like or dislike towards the product has been shown in [155]. They concluded that theta-band activity near frontal, parietal and occipital lobes are reflective of human preferences. [156] establishes the feasibility of detecting subjecting preferences through N200 signals, LPPs, and Positive Slow Waves (PSWs). Moreover, the authors found that subsequent buying decisions also modulated the LPPs. [157] reported an average accuracy of 60% when predicting the preferred product from a pair of products using N200 and theta wave features. [158] classified 30 pairs of shoes successfully in two classes (buy and no-buy) for 40 participants. [159] developed a predictive modeling framework to understand consumer choices towards e-commerce products from 14 categories (3 products each). An accuracy of 70.33% was achieved for the

consumer choice classification task using S-Golay filtering, Discrete Wavelet Transform (DWT) coupled with Hidden-Markov Models (HMMs).

2.5 Detection of Error-related Potentials

Error-potentials in EEG signals are studied under two paradigms in human-machine interaction tasks, (i) *feedback and response ErrPs:* error made by humans [160], (ii) *interaction ErrPs:* errors made by machines in interpreting human intent [161, 162]. As an instance of interaction ErrPs, [163] uses ErrPs in-tandem with P300 to boost the performance of the BCI speller device. Another interesting paradigm is when a human is watching and silently assessing a system. Several works propose the use of ErrP from a passive (or silent) human observer as feedback to a learning system. In [48], a simple robotic system that performs a binary selection task using ErrP as feedback is studied both in open and closed-loop settings. This enables ErrPs to be used as a supplementary reward for the Q-learning [164] or deep Reinforcement Learning (RL) algorithm. With the recent developments in deep learning, ErrP has also found application in reinforcement learning where it can be used as a *reward function*. [48] uses ErrP as a reward signal while a user is observing a robot perform a specific task. The use of error-potentials in human-computer interaction tasks, or for the acceleration of RL algorithms is underpinned upon the accurate detection of the error-potentials. Several approaches have been proposed in the literature to decode the error-potentials. [165] demonstrated the possibility of continuous and asynchronous detection of ErrP, while [161] proposed a statistical classifier. The state-of-the-art error-potential decoding algorithm relies on the Riemannian geometry framework and was proposed by Baranchant et al [166, 167]. It was later successfully applied for various classification paradigms in BCIs, namely, motor imagery, P300, SSVEP, etc. We explain the above algorithm later in this thesis, and provide comparisons with the proposed modifications in the algorithm to boost the accuracy.

Recently, there is a long line of papers studying reinforcement learning from human

feedback, such as [168, 169, 170, 171, 172]. However, they are only about explicit human feedback or labeling, and they all assume human feedback is noiseless. In this thesis, we perform reward-shaping using implicit human feedback, and also propose a practical framework to use reward function learned by imitation learning to augment the following RL agent. Numerous works [160, 173, 174] have studied a high-level error-processing system in humans generating the error-related potential/negativity (ErrP or ERN).

Interaction, response, and feedback ErrPs have been heavily investigated in the domain of choice reaction tasks, where human is actively interacting with the system [175, 176, 177, 161, 178] and the error is made either by the human or by the machine. [179] demonstrated the use of ErrP signals in an interactive RL task when the human is actively interacting with the machine system. [161] explored the ErrPs when human is silently observing the machine actions (and does not actively interact). Works at the intersection of ErrP and RL [180, 48] demonstrate the benefit of ErrPs in a very simple setting (i.e., very small state-space), and use ErrP-based feedback as the only reward. Moreover, in all of these works, the ErrP decoder is trained on a similar game (or robotic task), essentially using the knowledge that is supposed to be unknown in the RL task. In our work, we use labeled ErrPs examples of very simple and known environments to train the ErrP decoder and integrate ErrP with Deep Reinforcement Learning (DRL) in a sample-efficient manner for reasonably complex environments.

CHAPTER 3

UNSUPERVISED DETECTION OF EYE-BLINKS IN EEG

EEG signals are quite vulnerable to distortions caused by other interfering electrical fields. Specifically, eye-blinks produce a very strong interfering electric field (as the retina and cornea form an electric dipole [181, 182]) severely impacting the signal-to-noise ratio (SNR) of recorded EEG measurements. The presence of eye-blink artifacts in the EEG signal leads to confused or possibly false EEG interpretations. Hence, the detection and removal of eye-blink components can be significantly useful in any EEG analysis. Several algorithms have been proposed in the literature to identify eye-blinks, but they are characterized by one of the following limiting requirements - (i) a partly manual inspection for thresholds or template selection, (ii) a user training phase, (iii) a high number of EEG channels, and (iv) Electrooculography (EOG) data requiring additional electrodes above and below the eyes.

In this context, we first show that the brainwaves generated when a user blinks are detectable with a high degree of robustness. We then propose a fully automatic and unsupervised (i.e. *without requiring any training from the user*) blink detection algorithm, *BLINK*, to identify accurate timestamps of eye-blinks in the EEG data. The precise time-stamping of eye-blinks in the EEG data maximizes the availability of clean EEG for analysis, and can provide insights into blink duration and blink interval. *BLINK* relies on the natural frequency of occurrence of eye-blinks to self-learn brainwave profiles for each specific user's blinks, and hence does away with any user training requirements. *BLINK* design requires only a single EEG channel to operate.

Through extensive user experiments we show that *BLINK* can detect eye-blinks robustly across different EEG headsets and various user activities. We use two different commercially available BCI platforms, Muse and OpenBCI, to show the generalizability of *BLINK* over EEG headsets. We use controlled and uncontrolled user studies to evaluate the per-

Table 3.1: EEG datasets collected for *BLINK* evaluation

Dataset	Device	Type	Users	Total	Activity
EEG-IO	OpenBCI	Involuntary	20	500	external stimulation
EEG-IM	Muse	Involuntary	20	500	external stimulation
EEG-VV	OpenBCI	Voluntary	12	750	watching video
EEG-VR	OpenBCI	Voluntary	12	600	reading article

formance of *BLINK* over involuntary and voluntary eye-blinks respectively. Overall, we collected 4 different user EEG datasets (Table 3.1) with real users containing more than 2300 eye-blink waveforms. We show that *BLINK* detects eye-blinks with an accuracy of over 98% for all four datasets along with a high degree of precision.

We have publicly released our collected datasets and code[1] for the *BLINK* algorithm so that the presented results can be reproduced[2]. To the best of our knowledge, this is the first ever annotated eye-blink EEG dataset released in the public domain. We later discuss a methodology for using *BLINK* as-is in an online fashion to enable real-time eye-blink detection. This can widen the applicability of *BLINK* in the domains of Brain-Computer Interface (BCI) based communication and control, and real-time EEG data processing.

3.1 Background and Detection Challenges

The science behind eye-blink artifacts in EEG

EEG is designed to capture cerebral activity, but as EEG records electrical activity over the scalp, other interfering electrical fields distort the EEG signals. All the interfering signals which are not of cerebral origin are termed as artifacts. These artifacts include eye blinks, movements, cardiac, muscle artifacts, etc. Eye-blink artifacts present a specific superposition waveform on the frontal EEG channels, which diminishes as one moves towards occipital positions. Here, we briefly review the science behind why and how eye blinks affect EEG.

On a very high level, eye blinks cause a change in the electric field around the eyes

[1]Dataset and codes are available at `https://github.com/meagmohit/BLINK`
[2]User data is anonymized to ensure the privacy

which propagates through the head and affects the frontal electrode signals. The retina (at the front of the eye) and cornea (at the back of the eye) are negatively and positively charged respectively, resulting in a potential difference across the eye and hence forms an equivalent dipole. A different view pertaining to the eye-dipole theory is that the contribution of the cornea is insignificant, and the source of electric charge is mostly potential difference across retina resulting in the dipole shifted towards retina (posterior to the eye) [183, 184].

This dipole is capable of generating its own electrical field, the largest of this electrical activity can be measured by placing electrodes above and below the eye which is EOG (electro-oculogram). As EEG signals are a potential difference between two points, EOG (or an electrical field induced by an eye-dipole) can not affect EEG signal if it is constant over time.

The first model assumes eye-blinks to be associated with upward ocular rotation [185, 186], which changes the orientation of the dipole resulting in a transient change in the electrical field generated by the eye dipole, and hence blink artifacts on the EEG [187]. However, various researchers observed that (i) distribution of electric fields of blinking, and vertical ocular rotation are significantly different [188], (ii) The upward ocular rotation does not hold true with each and every eye blink [189, 190, 191], which discard the theory that source of interfering electrical activity is upward ocular rotation of the eyeball.

The second model considers eyelid movements as the main source of interfering electric fields. The upper and lower eyelids act as sliding electrodes as they move across the eyeball, capable of changing electric field around eyes [192, 193]. This electric field is opposite in polarity to that induced by vertical ocular rotation in the same direction. Further, the experimental results confirming the electrical change over the eye with fixed eyeballs and moving eyelids supports the view.

Blink waveform characteristics

A typical blink waveform on the frontal EEG is visually similar to a trough waveform in the voltage-time domain. Figure 3.1 shows a snapshot of such waveform at frontal elec-

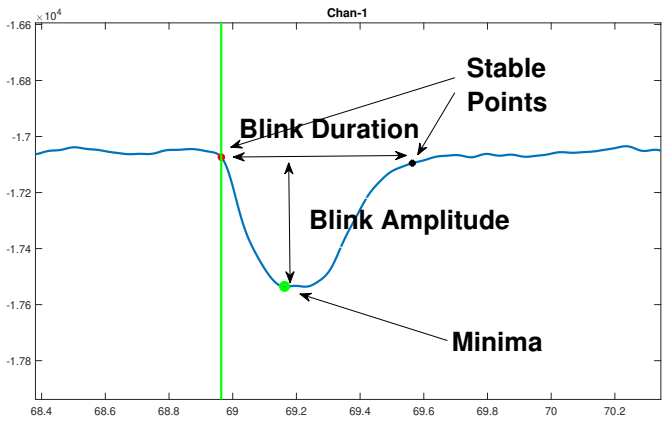


Figure 3.1: A typical eye-blink waveform

trode position (Fp1 in this case, according to the 10-20 electrode system) referenced to the earlobe electrodes (x-axis: time-domain, y-axis: voltage-domain). The blink waveform can be characterized by its (i) waveform pattern, (ii) blink amplitude, and (iii) blink duration.

A blink waveform pattern is defined as the voltage variation with time during a natural or forced eye-blink. The depth of the trough in the waveform pattern is known as the blink amplitude. Blink duration is simply the time taken by the user to perform the blink.

Detection challenges

Detecting eye-blinks is ostensibly easy as blink waveforms are visually prolific in features (as in Figure 3.1). The normalized blink waveform pattern (in time- and voltage- domain, i.e. single-unit time duration and single-unit voltage deviation) is consistent across multiple blinks of a single user, and also across different users. We can see this from Figure 3.2, that the similarity (correlation) of blink templates without considering amplitude deviation in correlation metric is similar for intra-subject blinks (multiple blinks of a single user) and inter-subject (blinks across users). In reality, state-of-the-art technologies present EEG waves inter-weaved with high-power noise (including inherent signal noise and measurement sensor noise). The variability across user-specific blink waveforms are so high across users (considering the amplitude deviation for blink waveforms) that if compared on the

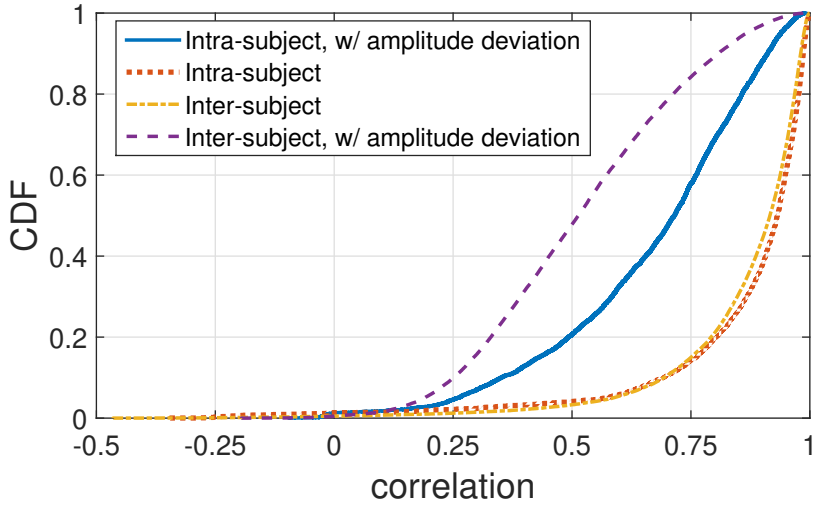


Figure 3.2: Correlation of eye-blink waveforms

same scale, what looks like a blink waveform for one user is simply noisy perturbations for another user. The high variability is not just limited to across users, but also is exhibited across different blink waveforms of a specific user (Figure 3.2 shows that when amplitude deviation is considered in the correlation metric, the correlation drops significantly in the case of blinks across users)[3]. This high variability among the blink patterns poses the first challenge of *designing a single universal algorithm that can account for the user and state variability, without an explicit requirement of fine-tuning algorithmic parameters.*

One might simply argue for the deployment of supervised training based approaches (e.g., neural networks, deep learning) to tackle the user-variability and noise issues like in the image or speech recognition problems. However, such a solution strategy is undesirable for wearable BCIs, where *user comfort* is an important consideration. Supervised training based approaches require users to go through an extensive training phase that directly impacts the usability and hence the consumer adoption of such devices. The second challenge, thus, is to *devise solutions that eliminate the user-training phase (essentially eliminating all supervised training based approaches*).

The above challenges when coupled with the *small form-factor constraints* (usage of

[3]For this result, we used *EEG-VR* dataset (Table 3.1)

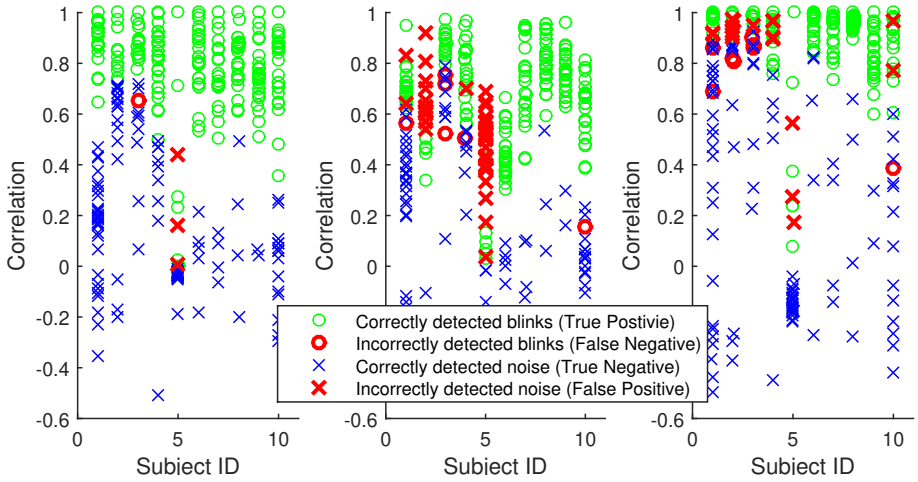


Figure 3.3: Correlation with template eye-blink waveform for given eye-blinks and trough-shaped noise
(i) template is constructed independently with amplitude deviation (intra-subject with amplitude deviation) , (ii) template is constructed together for all subjects with amplitude deviation (inter-subject with amplitude deviation), (iii) template is constructed independently without amplitude deviation in correlation (intra-subject)

fewer channels), and *high accuracy requirements with low false positives* (high precision - robust detection to avoid user frustration), considerably elevates the complexity of this problem. In summary, the key challenges in developing a blink detection algorithm are the following: (i) universality, (ii) no supervised training, (iii) small form-factor and (iv) accurate performance.

3.2 The *BLINK* Detection Algorithm

We propose an algorithm *BLINK* that is capable of robust blink detection without requiring any training from the user. *BLINK* is presented in Algorithm 1 along with subroutine 1.

3.2.1 Assumptions

BLINK operates on two assumptions

Assumption 1: Consistency of eye-blink patterns

It assumes that the eye-blink patterns are consistent for a single user for a short period (i.e., during data recording). However, no such assumption is made for different users (or

different recordings) and hence allows for user and session variability. To validate this assumption, we utilize the *EEG-IO* dataset (Table 3.1), which provides us with the timestamps of true eye-blinks. For the user EEG data with given eye-blink waveforms, we extract a template eye-blink signal (or fingerprint) based on the given eye-blinks, and compute the correlation of template with (a) noise waveforms (but similar to trough pattern) shown as *crosses* and (b) the given eye-blink waveforms shown as *circles* in Figure 3.3. Based on the correlation threshold comparison[4], if the waveforms are classified as eye-blink or noise using a threshold, we mark the corresponding incorrectly classified waveforms using red ink. The template extraction and correlation is done for users separately (total 10 subjects are shown in Figure 3.3, best-5 and worst-5 are shown) in Figure 3.3(i) and Figure 3.3(iii), and finally for all the subjects together i.e., one template eye-blink waveform for all users (global fingerprint) in Figure 3.3(ii). When subjects are treated separately, eye-blink waveforms can be assumed consistent i.e., a single template can represent all the eye-blink waveforms robustly and hence can distinguish from the noisy trough patterns. However, this is not true for multiple users due to the high overlap between eye-blink and noise correlation with the template, as in Figure 3.3(ii). Similarly, if amplitude deviation is not considered, the overlap between noise and eye-blink waveforms is significantly high, adversely affecting the detection performance (Figure 3.3(iii)). This establishes the consistency in eye-blink patterns for a particular user and can be leveraged to detect eye-blinks from the raw EEG feed efficiently.

Assumption 2: No other repetitive waveforms

There are no other repetitive waveforms in the input signal that present the same characteristics as an eye-blink waveform. This is a valid assumption, as frontal electrodes are mostly corrupted by eye-blinks, eye movements, facial muscles, and head movements. The pattern of other waveforms is either non-repetitive and random or dissimilar to the eye-blink

[4]A threshold was selected to minimize the number of incorrect classifications. For each waveform, its correlation was compared with the threshold to label as *eye-blink waveform* or *noise waveform*

Algorithm 1: *BLINK*[5]: an eye-blink detection algorithm based on feature detection and cluster-analysis

Input : E: EEG raw data, f_s: Sampling frequency
Output : $[t_{start}]$: start time of all eye-blinks, $[t_{end}]$: end time of all eye-blinks

1 Preprocess: lowpass filter E
2 $[t_{peaks}] \leftarrow peak\_detect(E, delta = 0)$
3 $[t_{start}], [t_{min}], [t_{end}] \leftarrow identify\_stable\_points(E, delta = 0, [t_{peaks}])$
4 **for** $i = 1, 2, \cdots, size([t_{min}])$ **do**
5 **for** $j = i+1, i+2, \cdots, size([t_{min}])$ **do**
6 $sig_a \leftarrow E[t^{(i)}_{start} : t^{(i)}_{min} : t^{(i)}_{end}]$
7 $sig_b \leftarrow E[t^{(j)}_{start} : t^{(j)}_{min} : t^{(j)}_{end}]$
8 $corr_{mat}[i, j] \leftarrow correlate(E, sig_a, sig_b)$
9 $power_{mat}[i, j] \leftarrow max(\frac{std(sig_a)}{std(sig_b)}, \frac{std(sig_b)}{std(sig_a)})$
10 $[index_{blinks}] \leftarrow high\_corr\_comp([[corr_{mat}]], [[power_{mat}]])$
11 $stable_{th}, delta \leftarrow blink\_typify([t_{start}], [t_{min}], [t_{end}], [index_{blinks}])$
12 $[t_{peaks}] \leftarrow peak\_detect(E, delta)$
13 $[t_{start}], [t_{min}], [t_{end}] \leftarrow stable\_points(E, stable_{th}, t_{peaks})$
14 Repeat *steps 5 to 15*
15 **return** $[t_{start}], [t_{end}]$

waveform (trough-shaped).

3.2.2 *BLINK* algorithm

Some properties of *BLINK* algorithm are, (i) *BLINK* relies on the natural frequency of occurrence of eye-blinks to self-learn brainwave profiles for each specific user's blinks, and hence does away with any user training requirements (it performs unsupervised learning); (ii) *BLINK* requires raw EEG data as input and returns the start and end positions of the blinks in the EEG data. Thus, *BLINK* can easily provide insights into the blink duration and blink interval; (iii) *BLINK* design requires only single-channel data. However, in the case of multiple channels the results can be combined to achieve more accurate results;

Algorithm Explanation: The pre-processing step (*line 1*) is to apply a low-pass filter to suppress high-frequency noise and smoothing the signal. The first step of the algorithm is

[5] [] and [[]] represents 1-D and 2-D array respectively in the algorithm, std represents the standard deviation

Subroutine 1: Subroutine *peak_detect* for *BLINK* algorithm

Input : E: EEG raw data, $delta$: threshold for peak detection
Parameters: w : size of the moving window

```
1 Initialize [t_min] with all local minimas in E
2 if delta is 0 then
3 |   return subset of [t_min] such that consecutive elements are separated by w units
  |     in time-domain
4 else
5 |   return subset of [t_min] such that consecutive elements are separated by delta
  |     units in voltage-domain
```

to find local minimas and stable points (Figure 3.1). Subroutine 1 (*peak_detect*) finds the local minimum points in the signal separated at least by $2w$ units in the time-domain (*line 2*). With each minimum point found, the algorithm searches for nearby stable points (*line 3*), where the signal fully recovers from the eye-blink trough (as shown in Figure 3.1). This is performed in function (*stable_points*) where the vicinity of each local minima is scanned to estimate the noise power (or $stable_{th}$), which in turn is used to compute aforementioned stable points such that the signal power from minima to a stable point crosses *stable_th*, but is limited after stable points. If, for any particular minima two stable points are not found (one on the left, and the other on the right), such local minimum points are discarded for further eye-blink investigation, and a set of stable points are returned for every other local minimum.

At this point (*line 3*), the algorithm has a set of trough patterns (each pattern consists of one local minimum and two stable points), which are further interpolated (as time length is different for each pattern) and linearly correlated on a one-to-one basis (*line 4-11*) to compute the cross similarity matrix in the time-domain (eye-blink shape) and the voltage-domain (eye-blink amplitude).

Further, highly correlated components of such patterns is computed (*line 12*) based on the time-domain similarity and a correlation threshold (which is kept low for robust detection) to find the matching repetitive patterns. The repetitive patterns might look similar (in the time-domain) but could correspond to eye-blink waveform (high amplitude) or simply

noise (less amplitude), which is further separated into two different clusters, and the high trough amplitude cluster is returned as potential eye-blinks. To make the algorithm more robust, resultant eye-blink patterns are profiled (smartly characterized) to have a better estimate of the noise power and the eye-blink amplitude (*line 17*, $blink\_typify$). Finally, a second pass is done to recover any missed eye-blink patterns (*line 14-16*), with the additional information of eye-blink SNR (signal-to-noise ratio) and user eye-blink profile. Thus, in the end, *BLINK* algorithm robustly detects all eye-blink patterns along with their start and end times.

Subroutine *peak_det* detects the minimas in the signal data separated at least by $2w$ units. The subroutine, if provided with a non-zero *delta* threshold, identifies the minimas which have at least of delta-amplitude difference with immediate maximas.

A careful inspection of the algorithm reveals that the parameters of the *BLINK* algorithm (and corresponding subroutines) are filter orders, different moving window sizes (time-domain), and correlation thresholds, which are not required to be tuned to different users, and thus allowing for the user-agnostic universality of the algorithm.

3.3 Evaluation

In this section, we first explain the user experiments conducted along with the correspondingly collected EEG data. We then evaluate the *BLINK* algorithm to validate its near-zero detection error with low false positives.

3.3.1 Experimental protocol and EEG dataset description

We have conducted four different user experiments to evaluate the robustness of the *BLINK* algorithm under a variety of EEG headsets and tasks. All the research protocols for the user data collection were reviewed and approved by the Institutional Review Board of the Georgia Institute of Technology. The subjects for the study were recruited from mixed demographics with an age range between 22 to 30 years old and were either full-time students or full-time employees. Upon arrival, the experimental protocol was explained

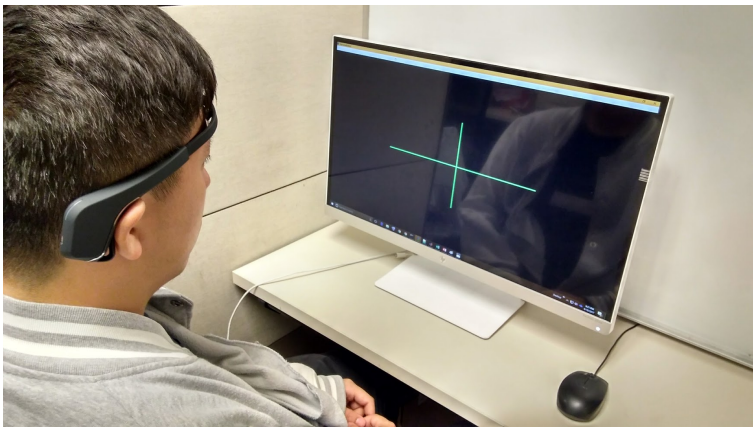


Figure 3.4: User evaluation setup

to the subjects, and the subjects were provided with consent forms and a demographic questionnaire. They were compensated with Amazon gift cards (10 USD value) for their successful participation in the study. The experimental paradigms and the collected EEG datasets are explained below:

A. Guided single eye-blink experiments: We collected raw EEG traces from 20 subjects in a guided (i.e., software instructed) environment where subjects were asked to perform a single eye-blink when instructed. Subjects were asked to sit comfortably in front of a computer screen and wear a BIOPAC 100C electrode cap [194]. Electrode gel was used to ensure the surface contact between the Fp1 and Fp2 (as per the 10-20 electrode system) electrodes on the scalp and forehead. Two silver ear-clip electrodes were additionally placed on the left and right earlobes to serve as a reference and to aid in the noise cancellation. The electrode cap was attached with the OpenBCI platform, which sampled the raw EEG at 250Hz. The digital signals were shipped to a desktop machine over the wireless channel. We used OpenViBE software (developed by Inria [96]) to present the on-screen stimulations and collect the user EEG data with synchronized timestamps. We also recorded a video of the subjects performing the experiments. The subjects were asked to perform a single eye-blink ONLY if a *green plus* appears on the screen (Figure 3.4). One experimental session presented 25 such external stimulations to perform eye-blinks every 3-4s depending on the subject's preference, resulting in the experiments lasting for

75 to 100 seconds per user. We repeated the same experimental protocol with Muse headset [195]. Muse headset is a dry-electrode headset and does not require a sticky gel to maintain the scalp contact. The Muse electrodes were moistened with water before the headset was worn by the user. We used the Muse Monitor application [196] on an Android platform to collect the user EEG data, however, the stimulations on a computer screen were still provided using the OpenViBE platform[6]. For both of the experiments, the video feed was manually reviewed, and true labels of the eye-blinks were marked for providing the *ground truth*[7]. These datasets collected from OpenBCI and Muse headsets were termed as *EEG-IO* and *EEG-IM* (Table 3.1), and were used to evaluate the performance of *BLINK* on involuntary eye-blinks and different EEG headsets.

B. Unguided eye-blink experiments: We also conducted uncontrolled user experiments with 12 subjects for the OpenBCI device where subjects were asked to (i) watch a video, and (ii) read an article, each for 5 minutes. These datasets were termed as *EEG-VV* and *EEG-VR* (Table 3.1). In unguided experiments, no external stimulations were provided. Other experimental and annotation methodologies were similar to the previous experiment. As the manual annotation process was time demanding, we annotated only the first 200 seconds of the unguided data, to use it for evaluating *BLINK* on voluntary eye-blinks and different user activities.

For all the collected datasets, ground truth, i.e., the annotation was performed before evaluating the *BLINK* algorithm to ensure an unbiased evaluation.

3.3.2 *BLINK* algorithm performance

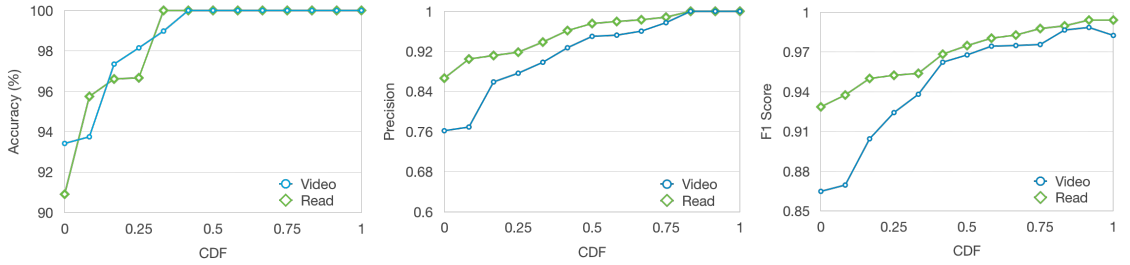
We evaluate the performance of *BLINK* algorithm using three different metrics. *Accuracy* measures the percentage of correctly detected eye-blinks out of total given eye-blinks (true positives). *Precision* refers to the number of correctly detected eye-blinks out of the total

[6]OpenViBE software does not provide the drivers of Muse headset for collecting EEG directly from the headset and hence, we used the Muse Monitor application.

[7]We performed the manual labeling as we found from the video feed that subjects blinked their eyes even when the green plus was not shown on the screen

(a) Accuracy **(b) Precision** **(c) F1 Score**

Figure 3.5: Detection performance results of *BLINK* algorithm on involuntary blinks



(a) Accuracy **(b) Precision** **(c) F1 Score**

Figure 3.6: Detection performance results of *BLINK* algorithm on voluntary blinks

detected eye-blinks. *F1 score* represents the harmonic mean of precision and recall. An ideal detection algorithm would perform with 100% accuracy, with precision and F1 score of 1 and 1 respectively[8].

The collected EEG datasets were analyzed offline by implementing *BLINK* algorithm (Algorithm 1) in Python. We analyzed the results for two frontal channels (Fp1 and Fp2) whose results were combined in an OR fashion. We used a 4th order Butterworth low pass filter (*algorithm 1: line 1*) with a frequency of 10 Hz. The w of subroutine 1 was set to $0.5f_s$, and w_1,w_2,w_3 of subroutine 2 were set to $0.5f_s$, $0.1f_s$, and $5f_s$ respectively, where f_s is the sampling frequency of EEG devices (250Hz for OpenBCI, and 256Hz for Muse). The correlation threshold for computing highly correlated components (*high_corr_comp*, algorithm 1: line 12), was kept to 0.2 (low value), to allow more potential eye-blinks for robust profiling.

[8]The detection problem is posed as detecting eye-blinks every time instant. It should be noted that the detection problem is not a binary classification problem, hence, the random baselines would not be 50% accuracy

Involuntary eye-blinks

We compute and present the detection performance of the *BLINK* algorithm on involuntary eye-blinks (i.e., *EEG-IO* and *EEG-IM* dataset from Table 3.1) in Figure 3.5 in the form of cumulative distribution for both platforms. The mean algorithm accuracy for all 20 subjects is near perfect (98.96% for OpenBCI, and 99.2% for Muse). The mean accuracy of (top-5, worst-5) subjects is (100%, 96.00%) for OpenBCI traces, and (100%, 97.2%) for Muse traces. The top-5 and worst-5 accuracies do not differ much, which validates the universality of the algorithm. Mean precision is above 0.9 for both the devices (0.951 for OpenBCI, 0.913 for Muse). Similar (top-5, worst-5) precision scores are (1.0, 0.858) for OpenBCI and (0.993, 0.801) for Muse. F1 score assigns a weighted score of accuracy and false positives. We received an average F1 score of 0.968 and 0.944 for OpenBCI and Muse, respectively, which confirms the robustness of the algorithm. Moreover, the results for Muse and OpenBCI do not differ much, which validates the extensibility of the algorithm across other BCI platforms.

Voluntary eye-blinks

EEG-VV and *EEG-VR* datasets (Table 3.1) were used to evaluate the performance of *BLINK* algorithm on natural eye-blink patterns when users were watching a video or reading an article. Figure 3.6 presents the performance of *BLINK* to detect involuntary eye-blinks in the form of cumulative distribution for both user activities. Averaged over 12 subjects, we achieved an accuracy of 98.4% and 98.3% for video and read activities respectively. The corresponding average precision measures and F1 scores are (0.92, 0.94) for video, and (0.95, 0.96) for reading activity. The consistent performance of *BLINK* on natural eye-blinks over different activities show the robust performance and applicability of *BLINK* in practical uses.

A summary of the *BLINK* performance is presented in Table 3.2 over the collected datasets.

Table 3.2: A summary of *BLINK* performance over collected datasets

Dataset	Accuracy	Precision	F1 Score
EEG-IO	98.96 ($\pm$ 2.32) %	0.950 ($\pm$ 0.062)	0.968 ($\pm$ 0.031)
EEG-IM	99.2 ($\pm$ 1.92) %	0.913 ($\pm$ 0.079)	0.944 ($\pm$ 0.046)
EEG-VV	98.47 ($\pm$ 2.44) %	0.922 ($\pm$ 0.083)	0.950 ($\pm$ 0.046)
EEG-VR	98.32 ($\pm$ 2.86) %	0.952 ($\pm$ 0.043)	0.967 ($\pm$ 0.022)

Table 3.3: Performance comparison with BLINKER

	Accuracy	Precision	F1 Score
BLINK	100%	0.952	0.97
BLINKER[28]	44.05%	0.558	0.69

3.3.3 Performance comparison of *BLINK* with related work

Comparison with BLINKER[28]

For comparing the algorithm performance with BLINKER[28], we look at the mean of accuracy, false positive rate and F1 score for 7 subjects in *EEG-IO* dataset. BLINKER requires long EEG traces, and runs successfully only on the dataset from 7 subjects, hence we use 7 subjects out of 20 for result comparison in Table 3.3). We can see the significant difference in eye-blink detection performance of *BLINK* and BLINKER (Table 3.3). *BLINK* performs perfectly (100% mean accuracy, 0.952 precision), but BLINKER[28] performs 44.05% accurate with the precision of 0.558.

Comparison with the basic threshold approach

While we know that threshold-based comparison approaches are highly ineffective, a curious reader might be interested in the merits of the proposed algorithm. Hence, for completeness, we implemented a naive statistical algorithm to detect eye-blinks (used frequently in EEG community [197, 198]) by comparing the signal variance (or standard deviation) with a threshold. For *EEG-IO* dataset of 20 subjects, the best threshold value was learned (which results in the highest F1 score), and the corresponding accuracy obtained was 6.83%, precision being 0.441 with an F1 score of 0.66.

Table 3.4: Performance comparison with learning approaches

	Accuracy	Precision	F1 Score
BLINK	98.15%	0.951	0.96
SVM	46.49%	0.559	0.69
k-NN	67.82%	0.664	0.75
basic threshold	6.83%	0.441	0.66

Comparison with learning approaches

Having previously established the inadequacy of learning approaches to detect eye-blinks for our solution (requirement of user training), we compare the *BLINK* performance with learning approaches, namely (i) SVM [199], and (ii) k-NN (k-Nearest Neighbors) [200] to establish a baseline. For this comparison, we use a moving window of $0.5 f_s$ length with a stride of $0.1 f_s$ to bucket the features as *eye-blinks* and *no-blinks* based on the given labels. We split the *EEG-IO* dataset in an 80:20 ratio for training and testing. For SVMs, the linear kernels were used, and the number of nearest neighbors was set to 5 for k-NNs. For SVMs, we receive an accuracy of 46.49%, precision of 0.559 and f1 score of 0.69. Similarly, for k-NNs, the obtained metrics are 67.82%, 0.664, and 0.75, respectively.

Reported performance comparison with the related work

After attempting to run codes released with previous works [28, 119], we concluded that every proposed algorithm is followed by the process of optimizing the algorithm parameters on their collected dataset. Hence in Table 3.5, we present the reported performance metrics of the selected related works (optimized on their collected dataset) along with their limitations and compare against the *BLINK* performance. We can see that although [119] and [106] report comparable accuracies, they have limitations of not being fully automatic, or requiring multiple EEG and EOG electrodes respectively.

Table 3.5: Reported performance and limitations of the related work

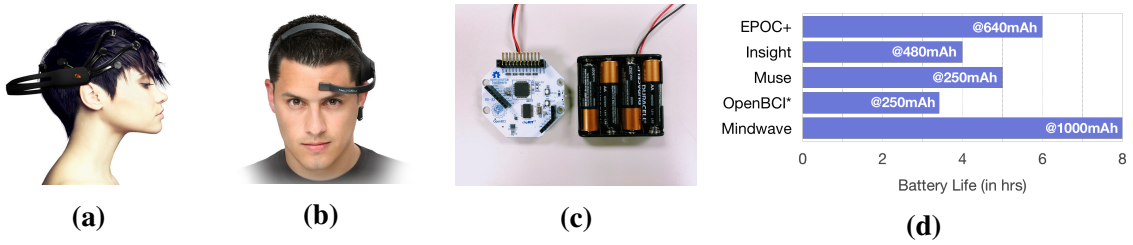
Algorithm	Performance	Limitations
[122]	87.13%	requires training phase
[119]	97.0% TPR (for 10% FPR)	not fully automatic
[106]	99.9% sensitivity, 94.7% specificity	uses 6 EEG, 2 EOG
BLINK	$\geq$ 98% accuracy	

3.4 Summary

In this work, we study the problem of eye-blink detection in EEG signals. In our literature review, we find that regardless of the abundance of research in this area, the applicability of the proposed algorithms is limited due to one or more requirements of multiple EEG channels, EOG channels, user-training phase and manual inspection for robust detection. In this context, we propose a fully automated unsupervised algorithm, *BLINK*, to detect eye-blinks in the EEG data. Our approach self-learns brainwave profiles for each specific user's eye-blinks, and hence does away with any user training or manual inspection requirements. *BLINK* capable of functioning on a single channel EEG accurately, estimates the start and end timestamps of eye-blinks very precisely. We collected four different EEG datasets to evaluate the robustness of the algorithm across various EEG headsets, user activities, and eye-blink types, and show that *BLINK* performs with an accuracy of over 98% in all cases along with an average precision of 0.934.

CHAPTER 4

LIGHTWEIGHT EEG-BASED WAKE-UP COMMAND DESIGN FOR BCI



(a) **(b)** **(c)** **(d)**

Figure 4.1: BCI wearable headsets and battery life (a) Emotiv EPOC+ , (b) Neurosky Mindwave, (c) OpenBCI system. In (d) we present the advertised battery life and battery capacity of currently popular BCI wearables in the consumer market.
OpenBCI system is also our experimental testbed where we implement the wake-up command detection and evaluate the system performance. (Images for EPOC+ and Mindwave headsets are obtained from `https://www.emotiv.com/epoc/`, and `https://store.neurosky.com/` respectively.)

EEG-based BCI platforms conform to a typical architecture. The user wears an *electrode array* (typically ranging from 2 to 32 electrodes)[1]. The electrodes are flat metal discs that can sense the electrical activity on the surface of the brain that occurs due to the electro-chemical exchange of signals between neurons. The electrical activity, also referred to as brainwaves, change from one region of the brain to another and are in response to different types of brain activity that in turn correspond to what the user is feeling or thinking [201, 202]. Because of the inherent complexity involved in the processing of the brainwaves to extract meaningful information, very little processing actually happens on the BCI cap. The brainwave data is shipped over a communication link to the "computer" where they are interpreted to deduce the user's thoughts. The link, especially in consumer-grade commercial solutions, is wireless and typically uses Bluetooth Low Energy (BLE). This "sense-ship-(remote)compute" model has a significant implication on the energy consumption properties of the BCI headset, and hence its battery life. *Since the headset does*

[1]High density EEG sensor arrays can have up to 256 electrodes.

not know when the user will issue a command through brainwaves, it has to listen on a continuous basis, capture the brainwaves, and ship it to the computer, for remote interpretation. The always-on mode of functioning limits the typical BCI wearable battery life to only a few hours. At the same time, numerous studies have established that battery life is a dominant factor in how users rate their experience with wearables [203, 204, 205, 206, 207].

The advertised battery life for commercially popular wearable EEG headsets are shown in Figure 4.1(d) and compared to the total battery capacity in mAh [208, 209, 210]. The battery life of even a relatively simple 8-electrode cap, is less than 3.5 hours, requiring users to charge their headsets multiple times a day, which is undesirable and severely impacts usability [211, 212, 213]. We believe that a longer battery life between consecutive recharges can be a critical feature to the end-user [206, 207]. Note that for non BCI wearables, the problem of battery life is heavily impacted by the display, and hence solutions tend to focus on intelligently switching off the display when not in use [214, 215, 216]. However, BCI headsets do not have a display and require a different solution to extend battery life.

Thus, in this chapter, we tackle the battery life problem for the BCI headset. We present the design of a wake-up command for BCI that allows the headset to operate by default in a near-sleep mode, and transition to a normal mode only when the user issues the wake-up command. The key challenge that we address is how the headset can operate in a near-sleep mode, but yet reliably detect and interpret a wake-up command (based on brain activity) from the user. Toward addressing the challenge, we pursue a solution strategy that is built upon the user's eye blinks.

We rely on three different user EEG datasets collected (Table 4.1) to evaluate and validate the performance of the *Trance* algorithm. We also implement the *Trance* algorithm on OpenBCI and demonstrate the detectability and power-requirements of *Trance* (in a resource-constrained environment). We have made the source code for the implementation

Table 4.1: EEG datasets collected for *Trance* evaluation

Dataset	Blink type	Users	Activity
EEG-MB	Involuntary	16	external stimulation
EEG-VV	Voluntary	12	watching a video
EEG-VR	Voluntary	12	reading an article

and an anonymized version of the dataset publicly available[2]. We experimentally validate that for typical active usage rates of wearables (2%, [217]), *Trance* can extend battery life by approximately $2.7x$, or to approximately 10 hours, allowing the headset battery to last for practically an entire day of use.

In summary, the following are our contributions,

- We perform a micro-power analysis on the individual components of a typical BCI wearable device, to identify the components that consume significant battery life[3]. We believe that such an experimental analysis could be useful for system designers.
- We design a new wake-up command that relies on eye-blinks as the command modality. *Trance* enables the detection of wake-up commands in a heavily resource-constrained environment through simple signal processing techniques.
- We perform an array of system and user experiments to evaluate and validate the system and the wake-up command performance. We study the interaction consequences through the lens of end-user usability.
- Finally, we provide a pointer to the data sets and implementations presented in this chapter. We believe that this will allow fellow researchers in the area to reproduce the findings and to build upon our contributions.

[2] `https://github.com/meagmohit/Trance`

[3] The detailed experimental power analysis of wearable BCI headsets is presented in the appendix (section 4.10)

4.1 Motivation

We perform a detailed experimental analysis to verify that (a) there is a limited battery life problem with BCI headsets, (b) there are meaningful control knobs to improve battery life, and (c) those control knobs are tunable to the optimal settings by using a wake-up command. We present the entire experimental methodology and analysis in the appendix section (section 4.10) and outline the salient learning below,

There is a limited battery life problem with BCI headsets:

We verify with the power experiments that a typical wearable BCI headset battery life is 3.4 hrs. The experimentation involved the average current measurement and approximate battery life projection by assuming the constant voltage.

Control knobs are available to improve battery life:

- We identify six different control knobs i.e. reconfigurable micro-components of the BCI hardware which could have a potential impact on BCI battery life, namely (i) micro-controller (μC) clock rate, (ii) ADC clock rate, (iii) ADC channels, (iv) data rate, (v) Programmable Gain Amplifier (PGA), and (vi) radio module. Based on the datasheet based power-impact analysis and allowed reconfigurability, we eliminate three control knobs - ADC clock rate, data rate and radio module.

- We run an exhaustive experimental study of all combination of settings of remaining three control knobs, (i) μC clock rate, $\mathbf{f}$, (ii) number of ADC channels, $\mathbf{c}$, and (iii) programmable gain, $\mathbf{g}$. We measure power for a specific ($\mathbf{f_i}$,$\mathbf{c_j}$,$\mathbf{g_k}$) in an exhaustive manner from, $\mathbf{f_i} \in \{48,40,30,20,10,6\}MHz$, $\mathbf{g_j} \in \{24,12,1\}$ and $\mathbf{c_k} \in \{8,7,6,5,4,3,2,1\}$, and conclude that PGA does not significantly impact the battery life.

- For the uC clock rate and ADC channels, we capture their contribution to power consumption in the form of a linear equation.

The case for wake-up command:

(i) we show that it is possible to achieve over 10 hrs of battery life for a BCI wearable

if the impactful control knobs are tuned down to their lowest setting, when the headset is not being used and (ii) the main challenge that remains is to reliably detect the wake-up command in the lowest parameter setting of the BCI headset (low CPU frequency, and sampling only a few electrodes).

4.2 Rationale for Using Eye-blinks

The first issue we tackle in designing the wake-up command is the choice of the basic building block, or modality, for the command. For e.g., 'Amazon Echo' and 'Google Home' harness natural voice (or speech) as their command modality. We build the foundation of our command solution in this work on *eye-blinks*. Alternative control modalities have been proposed for wearable computers. The requirement of these modalities has been laid out in the relevant literature [218]. Building upon these existing works and our use case, we formally list out the desired properties of an ideal modality for the wake-up command - (i) it should be easy, comfortable, inconspicuous and natural for the users, (ii) it should require no external aids or stimulations (e.g., flashing strobes), and (iii) the impact on the EEG signal must be pronounced enough to be quickly and robustly detected in a low-power mode and hence easy to detect.

Schaffer et al. [219] highlight the importance of input performance, for modality usage. In [218], Calhoun et al. argue that the input device needs to be inconspicuous (thus, avoiding any negative social consequences) while being obvious, natural and should require little to less training. Simultaneously, it should be oblivious to the environmental factors, e.g., ambient noise, light, temperature, etc. Simpson et al. [220] reflects on the unwillingness of users to use the intrusive modalities attracting attention. Additionally, users tend to prefer modalities that avoid inconvenient interaction steps, even if it increases the interaction time [221, 222].

The key benefits of relying on eye-blink based command are as follows:

- *Signal consistency:* The act of eye blinking affects the EEG in a distinct manner as

compared to the other modalities. The opposite electric polarity between the cornea and retina essentially turns the eye into an electric dipole, distorting the electric field around the eyes. This electric field change captured at the frontal electrodes in EEG, manifests a consistent change in EEG, and thus makes it feasible to detect without any user-training and data-driven learning [223].

- *Absence of the hardware control:* A survey of off-the-shelf BCI headsets (e.g., Emotiv EPOC+, Insight, Muse, Mindwave mobile 2, Intendix Speller, Neocomimi, Mindflex, etc.) shows that the headsets do not readily come equipped with other input modalities like buttons or touch interfaces. Thus, relying on EEG and Eye-blinks which the BCI hardware is already equipped to support, is considerably more desirable from the standpoint of necessary hardware modifications.

- *Competition for the action:* In mobile scenarios (e.g., running, driving, etc.), users need to pay attention to the environment, and taking hand-based actions might be dangerous [224]. Eye-blink based command provides a convenient way to wake-up the BCI device in such scenarios.

- *Non-intrusive:* One of the central goals of the BCI wearable is to allow a non-intrusive way of communication between users and computers. Relying on button or touch, gestures, or natural voice disrupts the environmental state around the user. Huang et al. [144, 140] support the non-intrusiveness of eye-blinks as a communication modality.

The act of blinking can be performed without any external aid. Such qualities make eye blink a perfect fit for the command modality. We now provide a qualitative comparison with the other possible wake-up command modalities. The candidate space for the command modality can be broadly classified into two categories, (i) user-action based commands, and (ii) user-thought based commands.

Table 4.2: Preference for different wake-up command modality (in comparison to eye-blinks) over various design parameters

	Input Performance	Natural FPR	User Training	Hardware requirement	Cognitive effort	Competition for action	Intrusive
Tactile input	●	●	●	○	●	○	○
EMG (Facial/Jaw)	○	●	○	◐	○	◐	○
Gestures	◐	◐	○	◐	●	○	○
Natural Voice	●	●	○	○	●	◐	○

●: Preferred ○: Not Preferred ◐: Comparable or can't say

4.2.1 Comparison with user-action based modalities

Calhoun et al. [218] describes the hands-free input interfaces for the wearable devices. Within user-actions, we consider (i) tactile input (e.g., button or touch), (b) EMG based facial, jaw or head movements, (c) gestures (motion-sensor based), and (d) natural voice. Schaffer et al. [219], presented various factors considered by the users for input modality selection. We select multiple user- and system- based factors to provide a qualitative comparison for the preference of user-action based modalities against the eye-blinks in Table 4.2.

Tactile input provides the best input performance with the fastest task completion time [222, 225, 226, 227]. However, it requires hardware modification on the BCI wearables and is intrusive to the user-environment. Convenience to deliver command plays a significant role in user adoption in hands-free approaches [228] against button or touch modalities. Facial muscle contractions, raising an eyebrow, clenching the jaw are detectable through electromyography (EMG) sensors, which can also be picked up by EEG electrodes [229, 230, 231]. These qualities make EMG based muscle movement compatible with existing BCI headsets. They are inherently inconsistent in terms of the signal signatures across users and across time, even for a single user. Hence, true proportional control is difficult and requires training [218]. Such inconsistencies are typically addressed through sophisticated algorithms [232, 233] that cannot be accommodated by limited computational capabilities. Thus, we argue that such user-actions are also not firmly suitable candidates for the wake-up

command modality. Another issue is to select a body (or muscle) movement that does not interfere with the normal functions of the user or can be discriminated robustly against the inadvertent one. Additionally, the anticipated frequency of use must be taken into account, as frequent uses of jaw clenches can aggravate Tempero-Mandibular Joint (TMJ) disorder [234].

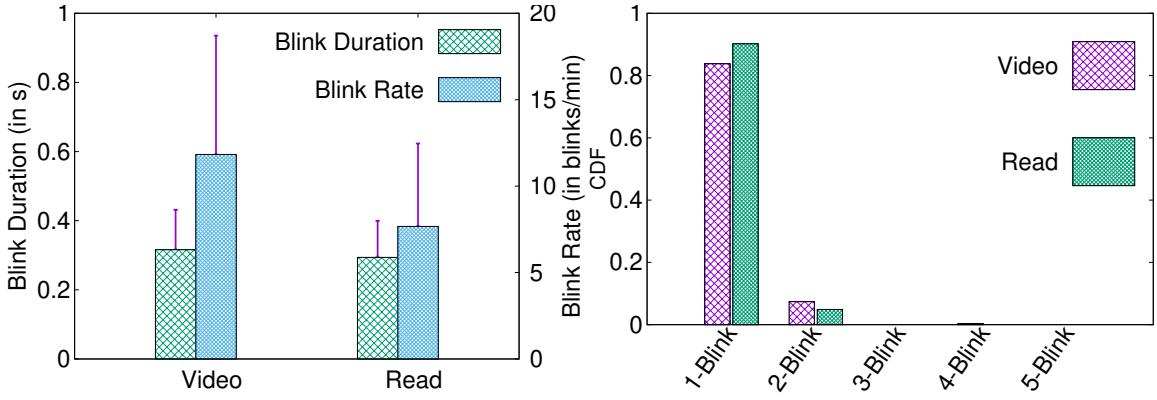
Existing BCI headsets are equipped with motion-based sensors (e.g., accelerometer, gyroscope), hence, compatible with detecting movement-triggered gestures. Kela et al. [235] suggested gestures as a natural modality for commands with a spatial association in design environmental control. Voice-based systems are the most natural way of human-computer interaction, as it is similar to the ways humans interact with each other [236]. They are easy to perform and present a comparable time for command delivery. For BCI headsets, the primary issue is the installation of additional hardware on the BCI headsets. They must perform in highly noisy and dynamic environments, and should not interfere with regular human communication. Considering privacy, speech or gestures may not be appropriate to use [218]. Noronha et al. [137] showed that users perceived eye-wink based modality at least as or more safe, easy and effective to use as the other modalities (i.e., voice, EMG gesture control) through subjective assessment and user questionnaires in a Human-Robot Interaction (HRI) task. Novanda et al. [237] found no significant difference between human efforts for completing a task in HRI over voice, touch and gestures. However, a significant difference was found in terms of human enjoyability, where touch as the input modality was least enjoyable for the users. Rudnicky et al. [222] showed users' strong preferences towards voice-based systems despite them being less efficient in terms of error and task completion time, over tactile input interfaces.

4.2.2 Comparison with user-thought based commands

In the context of BCIs, user-thought based commands can either be aided (or triggered) by an external stimulus (e.g., strobe light flashing at a certain frequency) or based on only thoughts (e.g., imagining limb movements). Any user-thought modality that is dependent

on an external stimulus will not satisfy the independence requirement i.e., users would not be able to issue wake-up commands unless the external stimulus exists in the environment. The detection of pure user-thoughts (e.g., motor imagery [238], P300, etc.) is heavily dependent on statistical learning methods due to the inconsistency in the features exhibited across the users. Hence, the detection of such modalities [239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249] demand extensive user-training and require highly sophisticated filtering and machine learning algorithms. The limited hardware capability in a typical off-the-shelf BCI hardware makes it infeasible to train and run such algorithms directly on the hardware especially when operating in low-power mode. This is in accordance with the detection latency of over 300ms [250] on a GHz scale machine. The latency of command detection in an MHz scale processor in the order of seconds, is not desirable for real-time detection. The buffering aspect in the continuous processing raises broader issues, when the detection time is more than the time the user takes to issue the command. Hence, such thought modalities (from the perspective of their state-of-the-art) are not practical for the wake-up command detection in a resource-constrained environment.

An Ultra-low Power Digital Signal Processor (ULP DSP) could be used to tackle the battery life problem in BCI headsets. If the ULP DSP were to support a thought-based wake-up command (e.g. motor imagery), the challenges discussed earlier would still remain significant - burdensome user-training, lack of consistency in signals across time, the computational complexity of the detection mechanism, and the need to sample from a large number of electrodes [251, 252]. While more exploration of this approach is needed, we believe that a ULP DSP system based on thought-based wake-up commands is unlikely to be easily realizable. On the other hand, if the ULP DSP were to be designed for use with eye-blinks, the system presented in this paper could serve as a candidate design for the implementation.



(a) Blink Statistics **(b) Multiplicity of Blinks**

Figure 4.2: Study of blink patterns

4.3 Trance: Wake-up Command and Algorithm

We now proceed to tackle the challenge of designing a wake-up command and a robust detection strategy, i.e., how BCIs can detect wake-up command in the resource-constrained environment. The next subsection explains the inefficacy of the single blink as a wake-up command and presents the design choice based on the multiple eye blinks.

4.3.1 Learnings from natural eye-blink patterns

According to the various studies [253, 254], it is estimated that a healthy adult blinks every 3-4 seconds. The blinking rate is highly variable across different people and tasks. In [254], Bentivoglio et al. state that the blinking rate is 17 blinks/min at rest, 4.5 blinks/min while reading, and 26 blinks/min while talking. We use the *EEG-VV* and *EEG-VR* dataset (Table 4.1) to study natural blink characteristics. We show the blink rate statistics in Figure 4.2a.

We also conducted uncontrolled experiments on 7 adult subjects to study such blinking patterns. We asked the subjects to (i) watch a video, and (ii) read an article, each for 5 minutes. During the experiment, we recorded EEG data and video feeds simultaneously. We later analyzed these recordings offline to locate the blinks within the EEG. From these experiments, it can be easily noticed that the natural blinking rate is very high. (8.57 blinks/min averaged on both activities). This is in accordance with our day-to-day

experience, and *thus a standalone single blink is an unfeasible candidate for the wake-up command.*

We analyzed the recorded data for blink duration and frequency of the multiple blinks. Figure 4.2a also shows the variation in blink duration. We notice that this deviation is high (standard deviation is greater than 30% of the mean blink duration), thereby restraining us from fiddling with blink duration for the command design. Figure 4.2b presents the cumulative frequency of multiple blinks. We labeled a group of single-blinks as multiple blinks if the gap was less than one second between the consecutive blinks. For e.g., three single-blinks were termed as "3-blinks" when the time gap between adjacent single-blinks was less than one second. It is evident from the above result that multiple blinks can be leveraged for the command design, which is researched in detail in the next subsection.

4.3.2 Wake-Up command design rationale

In the last subsection, we learned that the multiplicity of the blinks could be used as one degree of freedom for the command design to decrease the natural false positive rate. We consider an array of multiple eye-blinks based commands as the candidate space for wake-up commands, and analyze them in terms of their False Positive Rate (FPR) to select a default wake-up command. The natural FPR is the frequency with which the wake-up command will be detected due to the natural blinking pattern of the user, i.e., the user performs the wake-up command without any intention of using the wearable device. We study the natural FPR for video (*EEG-VV*) and read (*EEG-VR*) datasets (Table 4.1). For 2-blinks, natural FPR was 42.86 and 17.14 (per hour) for video and read tasks respectively. The natural FPR for 3-blinks dramatically reduces to 2.86 and 0 for video and read, respectively. Comparing the average natural FPR of 2-blinks (29.99 per hour) and (1.43 per hour), as also analyzed later in Figure 4.10, we select 3-blinks as our default wake-up command. In section 4.5, we conduct user studies to establish that 3-blink command is comfortable for the users to perform (Figure 4.8).

The default wake-up command presents a very low natural fpr, while being moder-

ately comfortable to the users. However, due to the individual differences between user preferences [255], we provide the users with the choice of switching to other multiple-blink commands. This enables the BCI wearable users to tune the wake-up command according to their natural blinking patterns, comfortability and performance (discussed in section 4.5). Hence, in the following sections, we provide a generic algorithm to detect k-blinks (k-consecutive eye-blinks) wake-up command and later evaluate the performance for $2 \leq k \leq 6$.

Design Goals: (i) *universality:* a single universal algorithm that can account for the user and state variability, and would not explicitly require training or fine-tuning, (ii) *small form-factor:* must function on one or two EEG channels, (iii) *lightweight:* the algorithm has to be simple (lightweight) and yet effective and should be able to operate in real-time (online) while relying only on limited hardware resources[4].

4.4 Trance Algorithm: Wake-Up Command Detection

We present our lightweight and online command detection algorithm, *Trance*, in Algorithm 2. *Trance* is a simple yet effective online algorithm, capable of detecting a series of blinks in the EEG data. In order to build an eye-blink fingerprint in an online fashion, *Trance* leverages the fact that the issued wake-up command will always have two or more consecutive blinks. *Trance* is built upon the robust noise handling and peak detection methodologies proposed in the signal processing literature [256]. *Trance* takes raw EEG data and the chosen wake-up command k as an input, and returns $True$ if the input data contains the k-blinks command.

It identifies the candidate blink signals using a peak detection methodology based on a threshold parameter ($delta$). These candidate blink signals are identified and validated

[4]As discussed in section 4.10, we use OpenBCI micro-controller unit (MCU) PIC32, in low-power settings as a representative for the target resource-constrained environment. PIC32 has a 128KB program memory size with a 32KB SRAM and 3KB auxiliary flash memory. In the low-power mode, the clock frequency of PIC32 is 6MHz.

[5][X] represents a set (an array) of elements $X^{(1)}, X^{(2)}, \cdots X^{(size(X))}$ where size(X) operation denotes the total number of elements in [X]

Algorithm 2: *Trance* Algorithm

Input : E: EEG raw Data , k: number of blinks in command, f_s: sampling frequency

Parameters: $delta\_init$: initial threshold for peak detection, inf: influence factor, $corr_{thresh}$: correlation threshold

Output : True if command is present otherwise False

Initialize: $delta \leftarrow delta\_init$, $found \leftarrow False$

Preprocess: lowpass filter (using moving average) E with cut-off frequency of 10Hz

$[t_{peaks}] = peak\_detect(E, delta)$

if $size([t_{peaks}])|^5 < k$ **then return** False; ;

$[t_{start}], [t_{min}], [t_{end}] \leftarrow$ identify_blink_candidates(E, $[t_{peaks}]$, $delta$)

$valid \leftarrow$ validate_blink_candidates(E, $[t_{start}]$, $[t_{min}]$, $[t_{end}]$)

if *not valid* **then return** False; ;

for $i = 1, 2, \cdots, size([t_{min}] - 1)$ **do**

  $corr \leftarrow$ correlate(E, $t_{start}^{(i)} : t_{min}^{(i)} : t_{end}^{(i)}$, $t_{start}^{(i+1)} : t_{min}^{(i+1)} : t_{end}^{(i+1)}$)

  if $corr$ ¿ $corr_{thresh}$ **then**

    $blink_{amp} \leftarrow$ compute_amplitude(E, $t_{start}^{(i)} : t_{min}^{(i)} : t_{end}^{(i)}$, $t_{start}^{(i+1)} : t_{min}^{(i+1)} : t_{end}^{(i+1)}$)

    $delta \leftarrow blink_{amp} \cdot inf + delta \cdot (1 - inf)$

  else

    $found \leftarrow False$

return $found$

based on their unique characteristics (e.g., pattern, slope, etc.) when the signals recover from the blink trough. The consecutive blink signals are correlated to perform blink detection. Further, the threshold value ($delta$) is dynamically updated according to the amplitude of detected blinks to adapt for the future wake-up command detection. In this manner, *Trance* detects a pair of blinks, and groups $k-1$ consecutive pairs to detect k-blinks.

The parameters of this algorithm are (i) initial peak detection threshold ($delta_{init}$), (ii) influence factor (inf), and (iii) correlation threshold ($corr_{thresh}$). $delta_{init}$ initializes the threshold to detect local peaks (between minima and maxima). A low value of this parameter successfully works for all users and blinks, as the threshold value is updated with an inf factor with each successful detection of a blink pair. The correlation threshold controls the trade-off between the accuracy and the false positives. A very low value of this threshold provides near-perfect accuracy with high false positives. These parameters can be set and

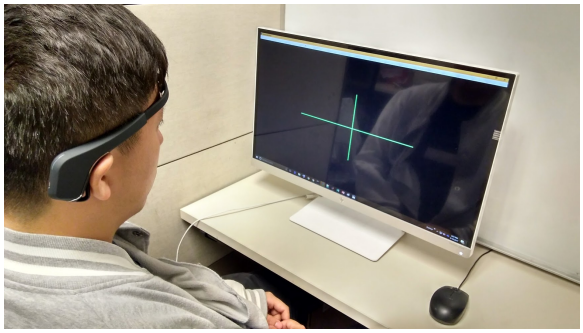


Figure 4.3: User evaluation setup

fixed offline as per the device noise level (during the device testing) and according to the required trade-off in detection performance, before releasing the firmware for use. *Trance* is agnostic to the user and state with respect to parametric changes, and thus is a universal algorithm. For the reproducibility of our work, we have released the dataset and code in the public domain[6]. For implementation and evaluation of *Trance* on OpenBCI device, the $delta_{init}$ parameter in the *Trance* algorithm was initialized to 200μV. The correlation threshold and influence factor were set to 0.6 and 0.05, respectively.

4.5 Evaluation

We conducted several system and user experiments to (i) evaluate the performance of the wake-up command and *Trance* algorithm, (ii) overall system performance in terms of latency and power savings, (iii) the broader interaction consequences in terms of user-comfort, time taken by the user to deliver the command and implications of false positive rate on the user-experience and system.

4.5.1 EEG-based user experiments

First, we conducted two EEG-based user experiments to evaluate the algorithms, wake-up commands, and prototype presented in this work. In this study, we decided to focus on two experiments, with one task for a controlled environment and two tasks for an uncontrolled environment, as it allowed us to study the user characteristics and assess the system

[6] https://github.com/meagmohit/Trance

performance in controlled and uncontrolled environments.

Participants

All the research protocols for the user data collection were reviewed and approved by the add Georgia Tech Institutional Review Board. A total of 20 subjects were recruited for the first task, and 12 subjects for the other two tasks. The subjects for the study were recruited from mixed demographics with a mean age of 26.75 years old ($\pm$ 2.17) and were either full-time students or full-time employees. 30% of the recruited subjects were females. All participants could communicate well in English and understood the experimental protocol. They were compensated with $10 Amazon gift cards for their participation in the study. The experimental paradigms and the collected EEG datasets are explained below,

Apparatus

For the EEG data collection, we used BIOPAC 100C electrode cap[7] The electrode cap was attached with the OpenBCI platform, which was further connected to a desktop machine over the wireless channel (using BLE). A Windows system (Dell Precision T3610) with a 27" monitor was used. We used OpenViBE software (developed by Inria [96]) to present the on-screen stimulations and collect the user EEG data with synchronized timestamps. A Logitech webcam was used to record the video of the subjects performing the experiments. We used *Flashback Express*, a screen recording software, to record the screen output along with the webcam output.

Task and stimuli

In the first task, the raw EEG traces were collected from 20 subjects in a guided (i.e., software instructed) environment. Subjects were asked to perform multiple-blinks when instructed. A *green plus* marker was shown to guide the user to perform two sets of triple-blinks with a small gap in between i.e., 3-blinks followed by 3-blinks. The frequency of the *green plus* was once in every 15-25s, and a total of 10 such stimulations were provided.

[7] https://www.biopac.com/product/eeg-caps-for-cap100c/

In the second task and third task, twelve subjects were asked to (i) watch a video, and (ii) read an article, respectively. The duration of each task was five minutes. While users were watching the video and reading an article, their EEG data was collected and the video feed was recorded. Users were asked to select a video and reading article of their choice, which would take at least 5 minutes to watch or read, respectively. Uncontrolled user experiments were conducted for 12 subjects to study the natural blink characteristics and test the natural and *Trance* false positive rate in such an uncontrolled environment.

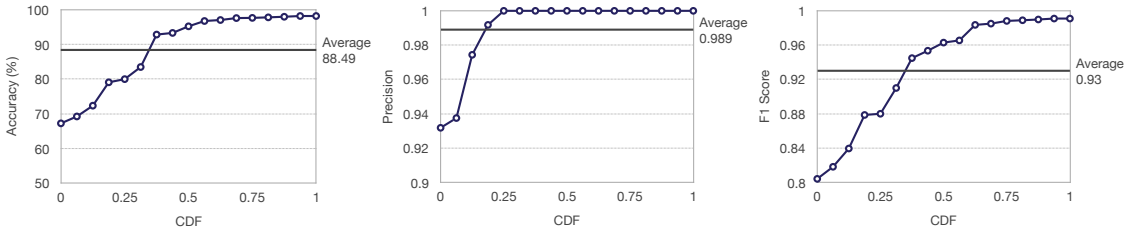
Procedure

Upon arrival, the experimental protocol was explained to the subjects, and the subjects were provided with consent forms and a demographic questionnaire. Subjects were asked to sit comfortably in front of a computer screen and wear the electrode cap. Electrode gel was used to facilitate the surface contact between the Fp1 and Fp2 (as per the 10-20 electrode system) electrodes on the scalp and forehead. After setup, an OpenBCI GUI software was used to verify the signal quality manually. Task-specific applications were initiated along with the camera feed and screen recording. After the completion of the experiment, users were asked to take off the electrode cap.

For both experiments, the video feed was manually reviewed, and true labels of the eye-blinks were marked for providing the *ground truth*[8].

For the first task, upon analyzing the video feed, we rejected the dataset of 4 subjects due to excessive head movements (essentially corrupting the EEG data), or improper placement of the electrodes for the controlled experiments. We term this EEG dataset of 16 users with multiple-blinks in controlled environment as *EEG-MB* (Table 4.1). For the second and third tasks, no external stimulations were provided, hence, manual annotation was done through the video feed. As the manual annotation process was demanding, we annotated only the first 200s of data for the evaluation. We term datasets obtained from

[8]We performed the manual labeling as we found from the video feed that subjects blinked their eyes even when the green plus was not shown on the screen



(a) Accuracy **(b) Precision** **(c) F1 Score**

Figure 4.4: Detection performance of *Trance* on default wake-up command (3-blinks)

these two tasks as *EEG-VV* and *EEG-VR* (Table 4.1).

4.5.2 User comfortability survey

We performed an experimental survey to study the user-comfort level of eye-blink based wake-up commands. We prepared an instructional survey form on Qualtrics where we explained the motivation of the study, and instructions to perform the series of blinks. In the questionnaires, the participants were presented with three different blink patterns to perform, and rate them on a Likert scale ranging from 1 to 5 with 5 being extremely comfortable[9]. The three different blink patterns were chosen randomly from 1-blink, 2-blinks, $\cdots$, 6-blinks. The survey was designed to take less than two minutes to complete. To ensure that participants were paying attention (and performing the tasks), we included two validation questions, (i) the number of blinks the participant performed in the first question, and (ii) to re-rate its comfortability score. The participants were recruited through Amazon MTurk[10], and were each compensated with \$0.02 conditioned upon the successful pass of the validation questions. A total of 209 responses were received; we removed 21 responses, due to incorrectly answering the validation questions.

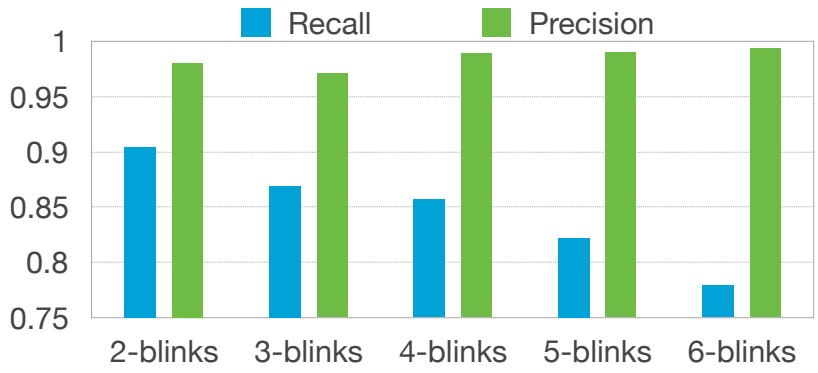


Figure 4.5: *Trance* performance on k-blinks wake-up command

4.6 Results

4.6.1 *Trance* algorithm performance

The performance of the *Trance* algorithm is evaluated using three different metrics, namely *recall*, *precision* and *F1 score*. *Recall* measures the percentage of correctly detected k-blinks out of the total given k-blinks. *Precision* refers to the number of correctly detected k-blinks out of the total detected k-blinks. *F1 score* represents the harmonic mean of precision and recall. An ideal detection algorithm would perform with 100% recall, with precision and F1 score of 1.0 and 1.0 respectively.

Performance over the default wake-up command

The multiple-blink EEG dataset (*EEG-MB*, Table 4.1) was used to evaluate *Trance* algorithm on the default wake-up command (3-blinks) mode. The dataset contains the ground truth labels for multiple eye-blinks in the form of the timestamps of each single-blink. As our default wake-up command is defined as 3-blinks with consecutive blinks within one second, we mark the ground truth in a similar manner. Specifically, in the ground truth labels, we mark 3 single-blinks (with consecutive blinks happening within one second) as one wake-up command. We present the cumulative distribution of (i) accuracy, (ii) precision, and (iii) F1 score in Figure 4.4 for 16 subjects. The mean recall obtained for the default

[9]The five rating choices were- 1: Extremely Discomfortable 2: Slightly Discomfortable 3: Neutral 4: Slightly Comfortable 5: Extremely Comfortable

[10]`https://www.mturk.com/`

wake-up command detection is 0.89%, with (top-5, worst-5) subject mean being (0.97%, 0.74%). We obtain a mean precision of 0.99, with a precision of 1.0 and 0.967 for the top-5 and the worst-5 subjects. Similar results are obtained for the F1 score, i.e., 0.93 averaged over all subjects, and the top-5 and worst-5 F1 scores are 0.99 and 0.85. For the wake-up command, we can see that there are moderate user variations in the best-5 and worst-5 for all three metrics. The users can tune the wake-up command as per their comfortability and performance.

Performance over the k-blinks wake-up command

In Figure 4.5, we compare the recall and precision of k-blinks wake-up commands. The total false positive rate for a wake-up command is the sum of *Trance* false positive rate (per hour) and natural false positive rate (per hour). The 2-blinks command has the highest recall of 0.95, with a precision of 0.98. Recall decreases with an increase in k, as for detecting a k-blink command, *Trance* has to detect $k-1$ consecutive pair of blinks accurately. For the 3-blinks command, we obtain a recall of 0.87, which decreases to 0.86, 0.82, and 0.78 for 4-, 5- and 6-blinks respectively. We obtain a very high precision value for all k- commands, which indicates that the false positives are very rare in *Trance* based wake-up command detection. For 3-blinks, precision is 0.97, and ≥ 0.98 for the other wake-up commands.

4.6.2 System performance

We implement the *Trance* algorithm on the OpenBCI board (Software platform: Arduino, Coding Language: C) to experimentally verify the overall system performance in terms of (i) latency in command detection, (ii) memory requirements, and (iii) power implications. For this experiment, we modify the OpenBCI architecture to run at (6MHz, 2 electrodes), and to receive raw EEG trace from the computer via RFDuino, instead of the electrodes. The trace-based analysis enables the correct measurement and replication of results which would not have been possible if evaluated directly on the prototype.

Latency in command detection

We fed the OpenBCI board with 10s snapshots of collected EEG traces (from the guided experiments), and measure the time taken by the algorithm to declare *command* or *non-command* (absence of command). We start the timer as soon as the OpenBCI receives the last bit of externally fed EEG trace. We repeat this experiment for multiple snapshots of *commands* and *non-commands*. *Trance* takes an average of 121.4 ms ($\pm$ 19.06) to detect a *command*. Detecting a *non-command* is significantly faster (due to the multiple earlier exit routines), i.e., 24.13 ms ($\pm$ 17.4 ms). The quick blink detection in order of a few ms, enables the real-time operation without adding any detectable lag for users. Along with latency measurement, while passing randomly interspersed EEG traces, we also re-verified the correctness of the *Trance* algorithm on the OpenBCI board. Thus, *Trance* is certainly viable on a lightweight platform (in terms of both computational power and memory) to perform real-time command detection.

Memory requirements

The memory required by the *Trance* algorithm on the OpenBCI hardware is 106.71 KB as compared to the default OpenBCI firmware (94.36 KB) out of a maximum possible 128KB. The dynamic memory requirement of our program is 11.73 KB, which is also only a slightly higher (and feasible) than the default value of 11.23 KB. This shows that *Trance* memory requirements are only marginally higher than default OpenBCI firmware (due to the additional *Trance* code) and within the maximum capacity of OpenBCI architecture.

Power implications

We transfer a 40s trace of previously collected user data (corresponding to *Trance* performance) to an OpenBCI device running the *Trance* algorithm. The trace contains two wake-up commands (randomly picked from 10 available commands from each user) interspersed randomly in the interval of 40s. The rest of the trace contains the noisy (non-command) data randomly sampled from the specific user data. This trace is processed by the *Trance*

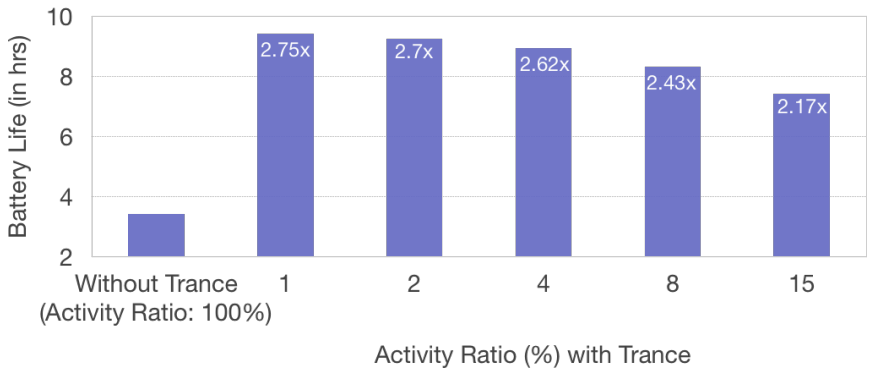


Figure 4.6: Battery life comparison for *Trance*

algorithm running on the OpenBCI (low-power mode) and generates the timestamps when the command is detected on the board. To measure the energy savings when using *Trance*, we run the OpenBCI device on low-power mode, and switch it to the high-power mode for a time duration corresponding to activity ratio (the percentage of the time, the wearable device is on high-power mode) for each detected command. We measure the average current drawn during the experiment duration (for different activity ratios) and compare it with the average current drawn in the absence of our solution (i.e., always in high-power-mode, 43.85mA). Figure 4.6 shows the battery life of OpenBCI for various activity ratios. With the power experiments, average current consumption over the users was found to be 16.22mA (9.3hrs for 2% activity ratio), experimentally verifying that with *Trance*, BCI wearables can last for single day usage. This compares to a theoretical projected lifetime of 11 hours for a 2% activity ratio. Liu et al. [217] establishes that wearable wake-up periods account for only 2% of the overall usage.

4.6.3 The study of usability

In this subsection, we look at the *Trance* solution through the lens of end-user usability. Specifically, we investigate (i) user-comfortability with the proposed wake-up command, (ii) time taken by the user to deliver the command, and (iii) false positive rate of the system.

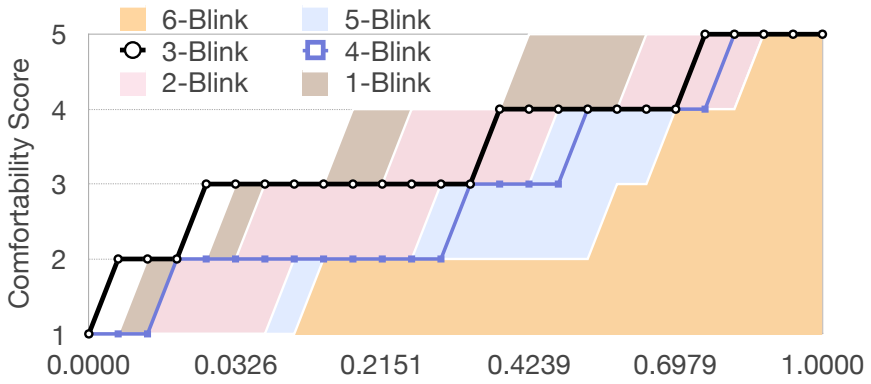


Figure 4.7: User comfort CDF over *k*-blinks

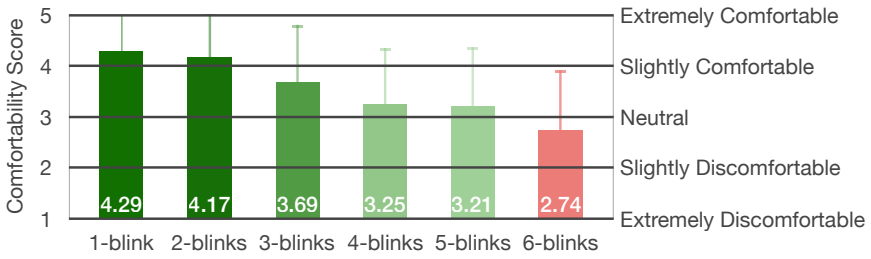


Figure 4.8: User comfort score over *k*-blinks

User comfortability

We use the Likert scale ratings from 188 valid responses collected in the user-comfortability survey. We present the cumulative distribution of 188 responses for each wake-up command in Figure 4.7. We also present the mean and standard deviation of the comfortability score of each wake-up command in Figure 4.8. 78.05% participants said that the default wake-up command (3-blinks) was not discomfortable. This compares to the 96.6% of participants, who did not find the 2-blinks command discomfortable. The average user-comfort score for 2-blinks was obtained as 4.17, a little higher than *Slightly Comfortable*. Similarly, for 3-blinks, we obtained a user-comfort score of 3.68, somewhat less than *Slightly Comfortable* but considerably higher than *Neutral*. For 4-blinks and 5-blinks, the comfortability score is very close to *Neutral*. For 3-blinks, 78.05% of participants did not find the wake-up command discomfortable. This compares to a similar statistic of 96.6% for the case of 2-blinks. We obtained a mean comfortability score of 3.69 ($\pm$ 1.11,

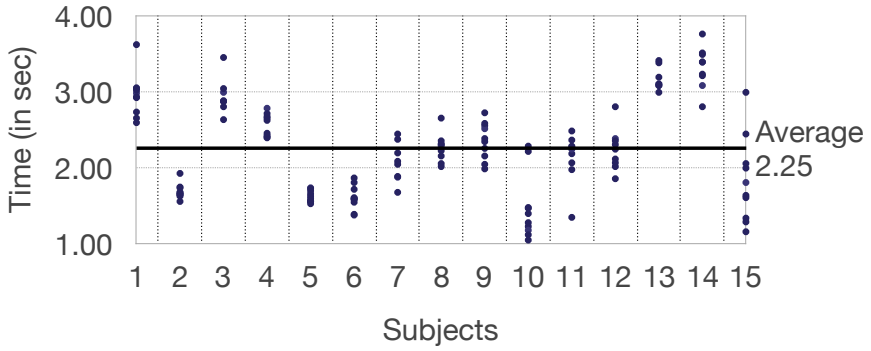


Figure 4.9: Action time over subjects

close to slightly comfortable) and 4.17 ($\pm$ 0.87) for 3-blinks and 2-blinks respectively. We performed the t-test on responses of two groups (i.e., 2-blinks and 3-blinks) and found the difference to be statistically significant ($p < 0.05$). This supports our intuition that the user comfortability in delivery of the default wake-up command (3-blinks) is less than 2-blinks. In summary, we found through survey-based user studies, that the wake-up command is reasonably comfortable to perform for the purpose of waking up the BCI wearables. We perform a t-statistic test to test whether the difference between the mean comfortability score of 2-blinks and 3-blinks is statistically significant. We obtain the p-value as 0.0015, rejecting the null hypothesis at the 0.05 significance level. In summary, the mean comfortability score of 2-blinks and 3-blinks was found statistically significant.

Time to deliver the wake-up command

For each trial, we measure the action time as the duration between the appearance of the stimulus (i.e., green cross) to the completion of the 3-blinks for the wake-up commands delivered in *EEG-MB* task. We present the action time for 15 subjects in Figure 4.9. Large variability is observed across subjects. Subject 10 took 1.47 ($\pm$ 0.43) seconds, while subject 14 took 3.31 ($\pm$ 0.28) seconds to deliver the command. Across all trials and subjects, a mean action time of 2.25 ($\pm$ 0.59) seconds was obtained to deliver the wake-up command. The command delivery time is comparable to the delivery time of other hands-free control modalities.

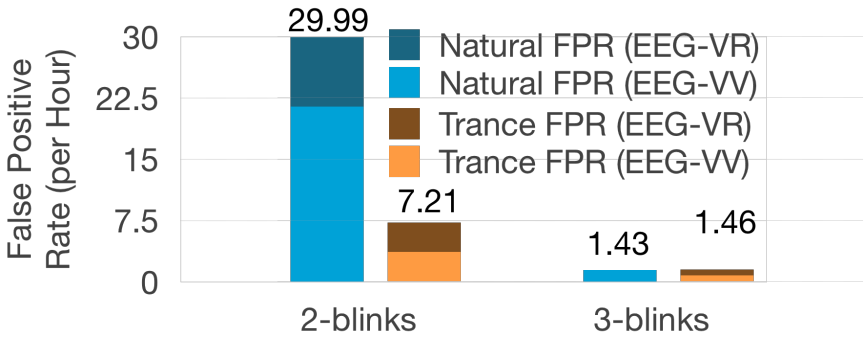


Figure 4.10: Natural and *Trance* FPR over k-blinks

False positive rate (FPR)

The false positive rate (per hour) of the system is defined as the frequency with which the wake-up command is detected without any user request to wake-up the BCI wearable. The total FPR for a wake-up command is the sum of *Trance* FPR (per hour) and natural FPR (per hour). The natural FPR is when the user issues the wake-up command as per their natural blinking pattern, without any explicit intention of waking up the device. *Trance* FPR is the result of *Trance* algorithm misinterpreting signals as the wake-up command. To evaluate both, we use the dataset from uncontrolled experiments (Table 4.1) when subjects were watching a video (*EEG-VV*) and reading an article (*EEG-VR*). We present the FPR in Figure 4.10. 2-blinks has the highest total FPR of 29.99 per hour (the natural FPR contributes 80.62% of it). With the increase in k, both natural and *Trance* FPR decreases. For detecting a k-blink command, *Trance* has to accurately detect $k-1$ consecutive pair of blinks, which results in a drop in the FPR. 3-blinks command performs accurately with a natural and *Trance* FPR of 1.43 and 1.46 per hour, respectively. A zero FPR (for both natural and *Trance*) was obtained for 4- or more blinks

4.6.4 Implications of the false positive rate

In the previous section, based on the experiment-based evaluation, we concluded that the proposed system performs with an FPR of 2.89 per hour. Here, we discuss the negative implication of this FPR. Firstly, an important thing to note here is that unlike other command

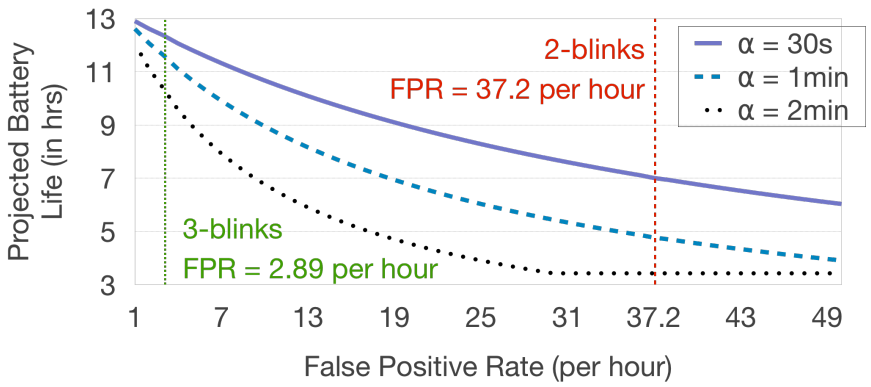


Figure 4.11: Implications of FPR on battery life

modalities, the FPR of the eye-blink based command modality does not have any negative implications on the user experience. In the case of a false positive, the BCI wearable will wake-up (i.e., switch to high-power mode) and wait for thought-based communication command from the user. If the system does not detect any ongoing communication, it will go back to sleep. Hence, a high FPR will have negative implications only on the battery life of the BCI wearable as the BCI wearable will keep switching to high-power mode needlessly. To quantitatively evaluate the impact of FPR on the battery life, we assume a simple scenario where the user is not issuing any wake-up command intentionally, i.e., we consider the scenario where the BCI wearables wake up either due to natural FPR or due to *Trance* FPR. We define a parameter α, as the duration of time BCI wearable will be awake (in high-power mode) before going back to sleep (low-power mode). In Figure 4.11, we show the projected impact of FPR on the battery life for the different awake duration (α). This curve is computed based on the current measurements obtained in low-power mode and high-power mode in (explained in section 4.10). In this scenario (α = 30 sec), the estimated battery life is 7 hrs and 12.36 hrs, for 2-blinks (total FPR = 2.89 per hour) and 3-blinks (total FPR = 37.2 per hour) respectively. Similarly, for α = 1 min, the battery life for 2-blinks reduces to 4.76 hours, while 3-blinks would last for 11.61 hours.

Table 4.3: Comparing proposed system with state-of-the-art wake-up command modality

	Natural voice (state-of-the-art)	Eye-blink (Proposed)
Recall	0.963	0.89
FPR (per hour)	0.11	2.91
delivery time (in s)	$\leq$ 2s	2.25 ($\pm$ 0.59)
processing time (%) per delivery time	24.82% (on 1.4GHz CPU)	5.39% (on 6MHz CPU)

4.7 Discussion

4.7.1 Comparison with popular wake-up command systems

To gauge the social acceptability of a novel wake-up command modality, we compare the proposed wake-up command system against voice-based wake-up systems (the widely adopted among the masses). We take Amazon Alexa as a representative example (with wake-word "Alexa") for comparison. We reviewed the testing performance of Amazon Alexa [257, 258] and compare it side-by-side with the proposed system in Table 4.3. Specifically, we use, *recall*, *false positive rate*, delivery and processing time of command. We can see from Table 4.3 that the recall and command delivery time is comparable. FPR for Alexa is very low (once every 9.1 hrs) as compared to the proposed system. However, we argue that the *Trance* FPR is acceptable and usable as it is not intrusive (no negative effect on user-experience). As discussed in the previous subsection, FPR presents negative implications only on the battery life of the system. In terms of processing time, the proposed system is very fast (takes 121ms on an average for 6MHz CPU) as compared to Alexa on a GHz scale processor. Translating on the same CPU scale, *Trance* performs an order of magnitude faster than voice-based wake-up command. Having said that, the proposed system can be considerably improved in terms of detection performance, interface design and usability. We believe that the presented system, along with the study and associated experimental analysis, could serve as a valuable baseline for the designers and researchers alike in this field of study.

4.7.2 Rationale for using OpenBCI as an experimental platform

This work is motivated with the examples of commercial BCI headsets (e.g., Neurosky, EPOC+), while the experimental and evaluation studies have been conducted on the OpenBCI platform. One might argue the disconnect between these BCI headsets, and hence, we provide the rationale for using OpenBCI as our experimental platform, and discuss the applicability of the proposed solution across BCI platforms.

1. While OpenBCI is a research-friendly BCI platform, it is also a genuine consumer-grade wearable headset that competes against the other commercial platforms [259]. Vourvopoulos et al. [260] compared OpenBCI with Emotiv, in terms of signal quality (classification accuracy) and usability (comfort, appearance, ease of setup), and found OpenBCI to be similar to that of EPOC+. Furthermore, in an effort to make the commercial adoption of OpenBCI easier, the platform's chipset system components for sensing biosignals (except the electrodes) are designed so that they can be coupled with any other commercially available electrode system or headsets (e.g., Ultracortex Mark IV [261]).

2. Second, the hardware architecture of the OpenBCI is quite representative of those of the other wearable headsets. Specifically, the three key control-knobs (uC clock rate, ADC channels, and wireless radio) that we rely on to make OpenBCI operate in low-power mode are all present in Muse [262], EPOC+ [263], and Neurosky [264]. Also, the signal quality provided by devices such as EPOC+ and Muse is rich enough for eye-blink detection [265, 266]. Hence, we are confident that the contributions in the paper are applicable to the other wearable headsets.

3. Finally, the critical reason that we did not use any of the other headsets as the experimental platform is that their firmware is not open source, and they do not have developer APIs to flash the firmware. The SDKs for Emotiv and Muse are available for developing applications, but not for firmware re-programming. While we could

have explored if reverse-engineering and hacking the firmware was a possibility, it was an ethical boundary that we did not want to cross.

4.8 Scope and Limitations

The context for this work is a scenario where the user wears a single EEG headset throughout the day (similar to smartwatches) and uses it to interact with multiple applications and tasks. By default, the EEG headset will be in low-power mode. The user would use the wake-up command to turn the headset on, before using it to issue an explicit command to an application. However, when the BCI commands are issued in the context of a specific application (e.g. a BCI-controlled text entry interface, a game or meditation program), the BCI would likely be active constantly while this application is running and disabled constantly (or not worn at all) while not, hence limiting the scope. The assumed scenarios do not accommodate all possible BCI applications and hence, its scope can be further refined. Briefly, the scope for the paper's contribution can be defined along four dimensions as follows,

1. User-capabilities: Trance applies only to scenarios where users are able to physically blink. Users suffering from conditions such as Eyelid Coloboma (where the eyelid is absent) will not be able to use Trance. Further studies have to be done to explore if Trance can be used by users suffering from other conditions such as Lagophthalmos or Bell's palsy disorders that cause weak blinks.

2. Input modality: Trance applies only to scenarios where the user is explicitly providing input using the BCI, i.e., *active BCI*. There are BCI applications where implicit input from the users is used (e.g., evoked potentials). For such applications, the BCI headset cannot go to sleep or low-power mode since the user does not actively issue the commands. For passive BCI uses (e.g., meditation), *Trance* requires an explicit wake-up signal, and thus, contradicts with the passive BCI paradigm. Trance will not apply for such applications. Further, input modalities that require the system to

present stimuli (e.g., SSVEPs, P300), the application can wake-up the device when the stimulus is shown, and hence, *Trance* will be irrelevant.

3. Frequency of use: Trance applies to scenarios where the user relies on the BCI headset with medium frequency. If the headset is either used all the time without any downtime (e.g., implicit input) or if used very infrequently (e.g., only two hours a day, in which case the user is more likely to put on the headset only as needed), Trance will be irrelevant.

4. Duration of a command session: Trance applies to scenarios where the command session duration is significant enough for a wake-up command to not become a disproportional burden for the user. If each command session lasts only for a few seconds (for e.g., to send an occasional command to the robot or another BCI-controlled device), the user might not want to incur the additional burden of having to issue a wake-up command.

The following are three example applications [with the four dimensions] that fit the above scope definition are-

1. Elderly assisted living [Capable, Explicit, 8-10 hours, 2-3 minutes] - Provide elderly persons more autonomy and independence by allowing them to complete otherwise difficult tasks through a thought.

2. High-consequence workplace training [Capable, Explicit, 2-3 hours, $>$ 15 minutes] - Leverage brain signals for high-consequence training to protect workers in high-risk jobs.

3. Brain based security [Capable, Explicit, 2-3 hours, 1-2 minutes] - Using brain signals for security including in authentication, non-repudiation, and identity-management.

Three example applications that do not fit the scope are,

1. Neuromarketing [Capable, *Implicit*, 2-3 hours, 4-6 minutes] - Leveraging brain signals to track user's reactions to market stimuli.

2. Neurogaming [Capable, Explicit, 4-6 hours, $>$ 15 minutes] - BCI used as the primary or secondary controller for users to interface with games.

3. Mindfulness [Capable, *Implicit*, 1-2 hours, 15-20 minutes] - Improving mental concentration and meditation with tracking brain signals.

Further, the scope of our work is restricted to the EEG-based BCI devices, and there are other BCI platforms (e.g. [267]) that may not fit this paradigm.

4.9 Summary

In this work, we propose a wake-up command detection strategy that enables always-on BCI platforms to run on low-power mode and transition to active mode only when the user issues the command, essentially solving the problem of charging BCI headsets multiple times a day. We use eye-blinks as the building blocks to solve the challenge of designing command, and detection strategy under the resource-constrained environment. Based on the user-characteristic analysis, we design a wake-up command for the BCI wearable headsets that balances the requirements of accuracy, false positives rate, and is comfortable for the users to use. We also present the lightweight *Trance* algorithm and through extensive experimental user studies, we validate the performance of *Trance*, and show that *Trance* can achieve 2.7x improvement in battery life.

4.10 Appendix: The Case for a Wake-up Command

4.10.1 BCI platforms

A typical BCI architecture consists of three core components: (i) an *electrode sensor array* placed on the scalp, (ii) a *hardware platform* to digitize, locally process and transmit

the brainwaves, and (iii) an *algorithmic processing platform* to analyze and decode the received brainwaves in an application specific manner. Scalp electrodes provide a conductive medium for the signal to reach the hardware interface. Wet electrodes require the application of gel to the electrode surface in contact with the scalp skin to reduce resistance, resulting in improved signal quality, but reduces usability. The hardware platform includes AFEs/ASICs (Analog Front-End or Application Specific IC) for digital sampling, ADC (Analog to Digital Conversion), and noise suppression. A wearable device worn by users embeds the first two components responsible for the acquisition, local processing and transmission of sensor data, and is also referred to as the "cap-end". The "mobile-end" serves as the algorithmic processing platform, and is typically either a smartphone or a computer.

Most of the commercially available BCI devices are application oriented e.g., Muse for meditation [11], Aurora for sleep analysis[12], etc. However, some of the devices are general-purpose and/or research grade devices e.g. Emotiv [13], OpenBCI [14]. Usually, BCI devices are evaluated in terms of their signal quality, usability index, form-factor, cost, etc. Compared to medical or research-grade BCI devices, wearable BCI devices are inferior mostly in terms of signal quality, but are cheaper and easier to use. A list of all the available consumer devices in the market that cost less than $1000 is available[15]. Several of the available BCI hardware either do not perform well in terms of available signal quality and usability, or provide severely restricted access to the system design and raw EEG data. Moreover, most of them only provide an SDK to develop applications at the mobile-end with a non-programmable hard-coded firmware at the BCI cap. We use the OpenBCI platform as the representative BCI hardware for our study as it bundles all the required features (transparent hardware design and software algorithms along with full access to raw EEG data) in a single piece of hardware. However, we also experiment with the Muse headband

[11] https://choosemuse.com
[12] https://sleepwithaurora.com/
[13] https://emotiv.com
[14] https://openbci.com
[15] http://www.autodidacts.io/neurotech-hardware-roundup-eeg-bci-tdcs-neurofeedback/

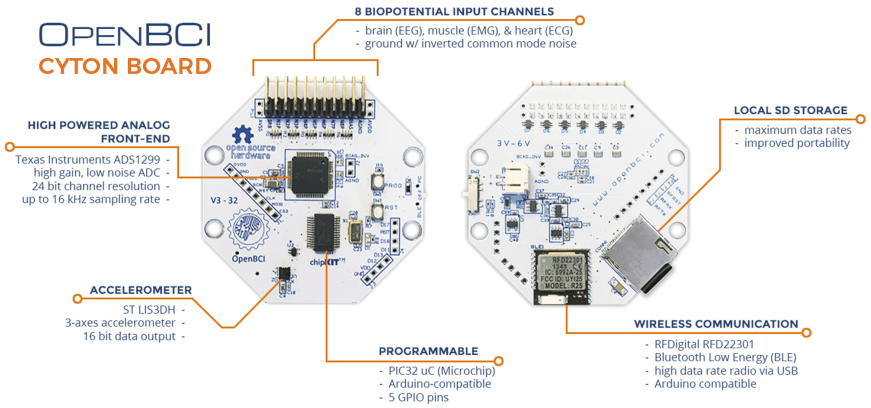


Figure 4.12: OpenBCI architecture
(Image taken from `http://openbci.com`)

to demonstrate the feasibility and extensibility of our analysis to the other available BCI platforms.

4.10.2 OpenBCI architecture

OpenBCI is an open-source, low-cost, programmable interface to access raw EEG signals. It has the capability to connect with upto 16 electrodes at a time, amplifying and digitizing the signals at 250Hz. As shown in Figure 4.12, the architecture of the OpenBCI board consists of three major components.

1. **Analog Front-End (ADS1299):** Designed and manufactured by Texas Instruments [16] for bio-signal measurements, this IC is responsible for digitizing and amplifying the EEG signals. It is a low-power, 8 channel, 24-bit ADC with built-in PGA (Programmable Gain Amplifier).

2. **Microcontroller (PIC32):** This Microchip PIC32 [17] Micro-controller is the central component of the OpenBCI board. It configures and coordinates with all the other ICs on the board to get data, arranges it, and transmits it to the radio module for forwarding to the "mobile unit". It is capable of executing instructions at 50MHz (default for OpenBCI is 40MHz). The program memory size and RAM is 128KB

[16] `http://www.ti.com/product/ADS1299`
[17] `https://www.microchip.com/wwwproducts/en/en557425`

and 32KB respectively. *PIC32 enables the local processing on the OpenBCI board.*

3. **Radio (RFDuino):** It is a finger-tip sized, low-cost, radio module, enabled with a μC to transmit the sensor data to the mobile-end through Bluetooth Low-Energy (BLE). The OpenBCI uses RFD22301.

Other components include an accelerometer (LIS3DH) and an SD card slot for 3-axis motion detection and external storage respectively.

4.10.3 Power analysis

Macro power analysis

The OpenBCI board, in its default development form, requires 4 AA (1.5V, 2300mAh each) batteries. However, using 4 AA sized batteries is clearly not suitable for wearable devices due to weight and safety considerations (as the platform is being worn on a user's head). Thus, we first perform power analysis on the platform as-is, and then extend the analysis for a typical wearable battery. Specifically, we use the battery specifications of an Apple Watch (250mAh, 3.8V, 0.94Wh) [18] and convert it into OpenBCI voltage requirements (6V, equivalent to 150mAh). To estimate the default battery life, we measure the current drawn in the hardware module and project the approximate battery life by assuming a constant voltage till the battery discharges [19]. Our experiments show that the battery life is only 3.42 hours. The advertised battery life for Emotiv EPOC+ (6 hrs on 680mAh) and Muse 2014 headset (5 hrs) also confirms our analysis of battery life for wearable BCI devices.

Micro power analysis

We now take a deeper look to identify the main source of power drain for a wearable BCI device. We identify the micro-components of the board along with their default settings, reconfigurability, and individual power requirements. We define "control-knobs" as those micro-components that are re-configurable inside the OpenBCI board, and could possibly

[18] http://www.onefruit.co/blog/2015/06/29/how-big-is-the-42mm-apple-watch-battery/
[19] We turn off the accelerometer for this particular analysis.

Table 4.4: Potential control knobs in OpenBCI board

Settings	Default	Configurable	Power
μC Clock Rate (MHz)	40	6 - 80	20mA (@3.6V)
ADC Clock Rate (MHz)	2.048	No	120μW
ADC Channels	8	1 - 16	-
Data Rate(SPS)	250	250 - 16k	-
PGA	24x	1x-24x	-
Radio (RFDuino)	ON	ON/OFF	11.8mA (@3V)

*(-) the information is not available in the datasheet

create a significant impact on battery life. We tabulate such micro-components in Table 4.4 [20] and explain them below.

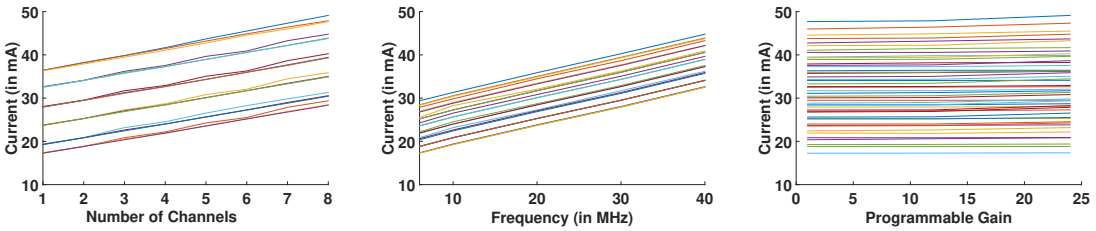
- μC/ADS Clock rate: This represents the operating frequency of PIC32 microcontroller and ADS1299 IC respectively, which directly affects their processing speed. As we can see from Table 4.4, the power consumed by the ADS clock oscillator is very low but high for μC (Remember from the previous section that OpenBCI draws an average current of 43.78 mA). So, we consider μC clock rate as one of our control knobs.

- ADC Channels: This denotes the form-factor of the device, i.e. the total number of channels from which EEG data is sampled simultaneously. Power consumption data per channel is not reported in the ADS 1299 datasheet, hence we consider this as our control knob for the power consumption analysis.

- Data Rate: The number of EEG samples recorded per second is known as the data rate. Following Nyquist Sampling Theorem, decreasing the data rate results in aliasing of the frequency components higher than half of data rate. However, in the case of OpenBCI, it is set to 250 SPS which is at its minimum value set by ADS1299.

- PGA (Programmable Gain Amplifier): PGA is an electrical amplifier with a control-

[20]The list is not exhaustive.



(a) vs #Channels **(b) vs. frequency** **(c) vs. PG**

Figure 4.13: Impact of different control knobs on current drawn, and hence power consumption

Table 4.5: Power analysis

Control Knob	Mean Power Deviation	Relative Importance
f	7.5554	62.90%
c	0.3804	3.17%
g	4.0757	33.93%

lable gain through external digital or analog signals. We consider PGA for the power consumption analysis.

Thus, for the power consumption analysis, our focus is on (i) μC clock rate (**f**), (ii) number of ADC channels (**c**), and (iii) programmable gain (**g**). The radio module will be turned off in the "low-power" mode, and hence we do not consider it for the power analysis.

To evaluate the impact of each parameter on the OpenBCI battery life, we run an experiment to measure the average current drawn (in mA at constant voltage) for a specific ($\mathbf{f_i}$,$\mathbf{c_j}$,$\mathbf{g_k}$) from, $\mathbf{f_i} \in \{48,40,30,20,10,6\}MHz$, $\mathbf{g_j} \in \{24,12,1\}$ and $\mathbf{c_k} \in \{8,7,6,5,4,3,2,1\}$. For each (**f**,**c**,**g**), we take 5 snapshots and average them to reduce the measurement noise variations, and repeat for all such possible ($\mathbf{f_i}$,$\mathbf{c_j}$,$\mathbf{g_k}$) i.e.a total of 142 data points. Once we have the average power consumed for all permutations, we define a metric "average power deviation" to evaluate the impact of each knob on the battery life.

For $\mathbf{f_i}$, we calculate the average power deviation over the other two variables as,

$$\frac{1}{|I|}\sum_{j,k} Var_i(\mathbf{f_i},\mathbf{c_j},\mathbf{g_k}) \tag{4.1}$$

i.e. we fix ($\mathbf{c_j}$,$\mathbf{g_k}$), and calculate the variance over all possible $\mathbf{f_i}$, and average over ($\mathbf{c_j}$,$\mathbf{g_k}$). We calculate a similar metric for $\mathbf{c_j}$ and $\mathbf{g_k}$ and report in Table 4.5 along with their percentage contribution.

Figure 4.13 shows the relationship of power consumed with each control knob. The colored lines in each plot represents the different possible values of the free parameters (e.g. f and PGA in Figure 4.13(a)). From the trend and relative average power deviation, it can be clearly seen that PGA($\mathbf{g}$) has a very low impact on battery life. Hence, we maintain its default value (i.e. 24x) to keep the signal quality unaltered. As the trend of power consumption is linear with both $\mathbf{f}$ (validates the PIC32 claim of 0.5mA per MHz power drainage) and $\mathbf{c}$, we fit a linear curve for power characteristics of OpenBCI,

$$\text{Current (mA)} = 0.4534 \times \mathbf{f} + 1.6615 \times \mathbf{c} + 12.8704 \tag{4.2}$$

The obtained R^2 statistic and p-values are 0.9994 and 0.0404 respectively for the above fit (eq. 4.2) which substantiates the goodness of the fit.

In the low-power mode ($\mathbf{f}$=6MHz,$\mathbf{c}$=1, radio=OFF) operation for 90% of the time [217], the estimated average current drawn will be 14.78mA, resulting in 10.14 hrs of battery life. This clearly shows that it is possible to achieve $3x$ improvement in the battery life provided the device is in the low-power mode when not actively used.

CHAPTER 5

TRACKING USER PREFERENCES USING BRAINWAVES

Knowledge of a user's preferences can be quite useful in several different contexts. For example, Amazon, the online retailer, sells over 600 *million* products. The Amazon landing page, on the other hand, can reasonably present only $50-60$ different products on a computer, and fewer on a mobile device. When a user arrives at the landing page, Amazon would ideally like to present those products that are of relevance to the user. Knowing the user's preferences at that point in time can help Amazon do so effectively.

Sophisticated user personalization models are routinely employed today by a retailer such as Amazon based on cues such as past purchases, searches, and items saved in cart. There are other contexts as well beyond online commerce where the ability to understand user preferences has significance.

Meanwhile, over the last couple of decades, rapid strides have been made in the domain of sensing and interpreting brain activity using electroencephalogram (EEG). Unlike its more involved counterparts such as magnetic resonance imaging (MRI) and functional-MRI, one of the distinct advantages of EEG is that the sensors can be used in a non-obtrusive user-friendly fashion. This advantage makes EEG a prime candidate for mainstream applications that reliably rely on brainwaves for understanding user thoughts. Advances in the understanding of brain architecture and functioning, coupled with sophisticated signal processing techniques, have allowed for EEG-based detection of user actions (e.g. blinks) and thoughts (e.g. motor imagery and error response).

In this work, we consider the intersection of the aforementioned domains. Specifically, we consider the detection and interpretation of user preferences using only the brain waves of the user detected using an off-the-shelf EEG wearable.

We consider this problem in the specific context of ranking a given set of objects based on a user's preferences. Thus, given a set of objects $OS = \{o_1, o_2, \ldots, o_N\}$, we consider

the problem of determining the respective ranks of the objects $RS = \{r_1, r_2, \ldots, r_N\}$, where $1 <= r_i <= N$, by only relying on the brain activity of a user who is wearing an EEG headset wearable. The following is a summary of our key contributions:

- Using an EEG dataset obtained from 14 users observing 10 different objects (products), we first establish the feasibility of object ranking based on an EEG wearable. We do so by relying on a brute-force trial and error based analysis of the EEG signals and comparing it to the ground truth of how the users explicitly ranked the corresponding objects.

- We then present a machine learning algorithm, *Cerebro*, that given a training set of EEG waveforms along with rankings from a specific user, can learn the specific nuances of the user's waveforms for preferences, and when provided with only the waveforms for a new set of objects can rank those objects accurately. The key novelty of *Cerebro* lies in the combined use of multiple aspects of the EEG signals (N200 mean, N200 minima, and Event-related Spectral Power (ERSP)) to rank objects according to user preferences, and a mechanism to self-determine when the algorithm's ranking is accurate enough to be actionable.

- We evaluate the *Cerebro* solution by training the algorithm with 7 objects for the 14 users, and evaluating the accuracy with which it ranks the remaining 3 objects as compared to the user-specified rankings.

5.1 Background and Problem Definition

5.1.1 User preferences

A user's preferences influence everything from mundane purchases to social behavior to moral decisions. The neurobiology of preferences is still an emerging area of study, but it is understood that preferences are influenced by both genetics and the environment. Since preferences heavily determine a user's actions, having visibility into the preferences can

help in several different scenarios. While we delve into some example scenarios later, we now briefly discuss some approaches to determine a user's preferences.

An obvious approach to learn a user's preferences is to ask the user for *explicit input*. For example, presenting a set of options to a user and having the user vote or rank on the options explicitly. An advantage of this approach is that the user's stated preferences are directly known. However, there are a few drawbacks: first, when user's share preferences they might not be entirely truthful and represent accurately their real preferences - this is observed routinely in pre-election polls; and second, since this approach requires explicit user involvement, it cannot be used frequently and for a large number of options.

An alternative to the explicit approach is to *implicitly observe* user actions and infer the user's preferences based on those actions. This is the preferred approach especially for environments such as e-commerce platforms where observing a user's actions is significantly easier than explicitly interacting with the user. A platform like Amazon observes a user's actions such as searches, clicks, time spent on a product page, additions to cart, and actual purchases to form a composite view of user's preferences and use this to appropriately optimize the options presented to the user. Video platforms such as Netflix and YouTube also rely on similar techniques to understand user preferences in order to present suggestions for users to watch next. YouTube's recommendation engine has a remarkably high success rate - over 70% of a user's watch behavior is directly from the recommended videos presented to the user [1].

There are some specific scenarios where it is neither possible for users to explicitly indicate preferences, nor is it possible to reliably track user actions to make meaningful inferences. For example, consider the problem of learning the preferences of a user with disabilities that preclude both explicit communication and any pertinent actions that would allow for meaningful inferences. Similarly, learning about the true preferences of young kids is a challenge.

[1] https://www.cnet.com/news/youtube-ces-2018-neal-mohan/

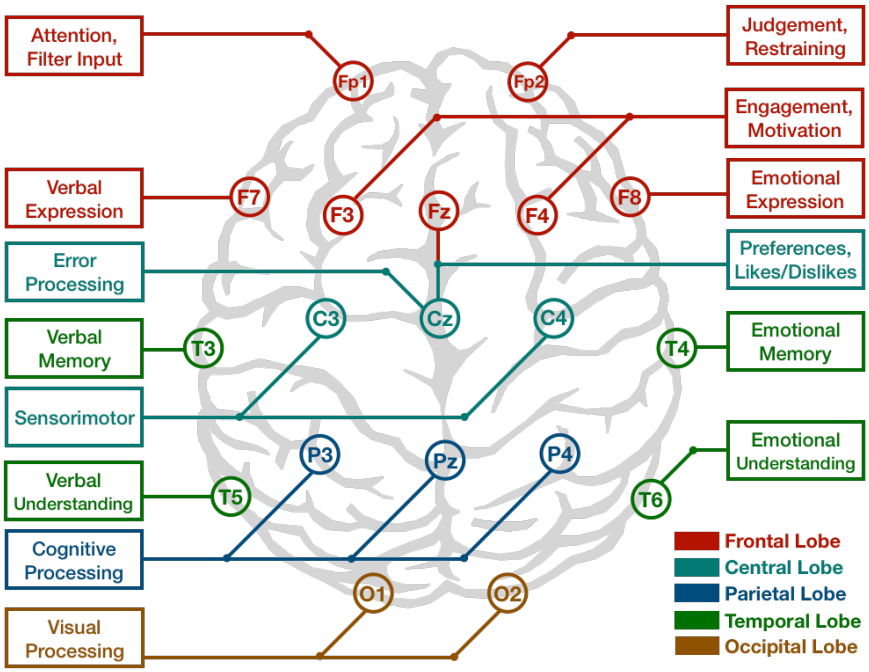


Figure 5.1: EEG electrodes and associated neural activity

In this work, we focus on *implicit observations, but not on the user's actions that can be somewhat infrequent, but on the user's thoughts*. Thoughts as a unit of observation are far more frequent, and more seamlessly accessible, than actions. Hence, there is considerable merit in considering the observation of thoughts using EEG in order to infer user preferences.

5.2 Target Scenario and Problem Statement

We consider a setup where a user is wearing an EEG headset while browsing through the e-commerce platform on her computer or phone. The electrode sensors continuously read raw brain signals, and the hardware platform transforms them into digital signals. The digitized brainwaves are transferred to the cloud over a wireless link for computational processing. The raw EEG signals are pre-processed (to increase the signal-to-noise ratio) and are dissected into fundamental frequency components (primarily theta and beta waves) in the cloud to search for specific patterns. The processed features are then subjected to learning algorithms to interpret their meaning. Thus, with such analysis, conscious or

subconscious user preference toward the browsed or recommended item can be inferred. If multiple objects are shown at the same time, attribution methodologies are required to tie user preference to a specific item. The user-specific model in the cloud is updated based on the learned preferences of the known item, which delivers the updated personalized recommendation to the user device.

An interesting and important aspect of this research is the incentive models for the user to wear EEG headsets in some specific use cases. A user incentive model is crucial when the targeted applications do not directly provide the innate value to the users. Targeted advertisements, personalized recommendation systems for e-commerce and digital media are a few examples that fall into such category. A trivial strategy is to provide rewards or monetary benefits (discounts) to users. However, providing additional benefits like assessing cognitive abilities, and understanding preferences while tracking focus and attention could be of more fundamental value to the users.

Our goal in this paper is to determine the preference ranking for a set of objects by only relying on the brain activity of a user who is wearing an EEG headset wearable. We define the mathematical formulation of the problem as follows,

Problem Definition: *Consider a user U presented with a set OS of N objects,* $OS = \{o_1, o_2, \ldots, o_N\}$. $S = \{s_1, s_2, \ldots, s_N\}$*, represents the corresponding recorded neural measures while a user is browsing objects from the set OS. There exists a ranking (or permutation) function* σ*, s.t.* $\sigma(o_1) \geq \sigma(o_2) \geq \cdots \geq \sigma(o_N)$ *in accordance with the preferences of the user. We explore the practical feasibility of designing an algorithm A, such that* $A(S, OS) \sim \sigma$. Specifically, in this work, we consider N to be 3, and present the ranking algorithm and its performance on ranking 3 consumer objects.

5.3 Feasibility of Object Ranking using EEG

5.3.1 Dataset

We rely on the dataset obtained through the experiments in [157] to perform our analysis. In [157], the experimental design involves a pairwise classification task where 14 subjects were shown 10 unique consumer products[2] and their neural activity was recorded simultaneously. Later in the experiment, the subjects were shown 2 random products side-by-side (out of 10) and were asked to choose and label the preferable product. The first part of the experiment was repeated 50 times for each product (per subject) and provided the raw neural signals. The latter part of the experiment included 45 unique product-pairs, and each product-pair was repeated 6 times to tackle the stochasticity in consumer preferences, which serves as ground truth labels for ranking and preference scores of the products (out of 54). In [157], the acquired EEG dataset was re-referenced to the ground electrodes (located behind the ear), and was filtered offline in the frequency band of 0.05Hz to 40Hz. For each external stimulus, epochs were extracted from the data for 800ms relative to the 200ms pre-stimulus baseline. Further, Independent Components Analysis (ICA) was employed to compute the independent components, and components corresponding to muscular and eye artifacts were manually removed from the signal to receive a cleaner signal.

5.3.2 Feature design

The source of the neural signals associated with the user preferences is known to be located in the fronto-central region. Hence, we performed the channel selection with F3, C3, P3, Pz, Fz, Cz, C4 electrodes according to the 10-20 electrode system. A cleaner ERP signal is obtained by decomposing the channel data in the independent components and obtaining the top component through the FastICA algorithm (Figure 5.2). The top part of Figure 5.2 shows the data from selected 7 channels, and the top independent component (IC-1). For simplicity, we will use the term *waveform* to mention IC-1 component of the EEG signal.

[2]In the conducted experiment, the objects were consumer products. In the rest of the work, we use the terms products and objects interchangably

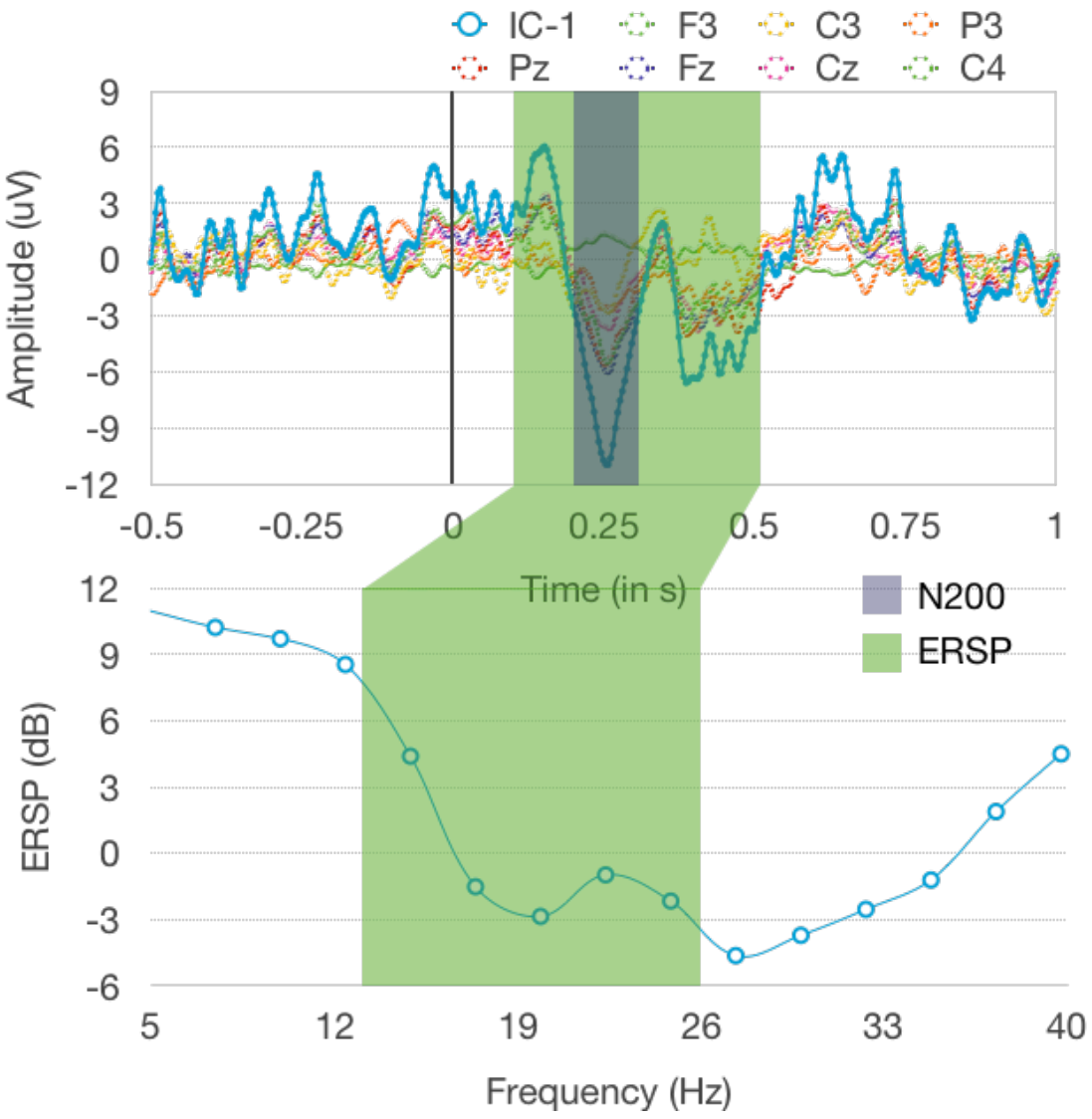


Figure 5.2: EEG waveform and features

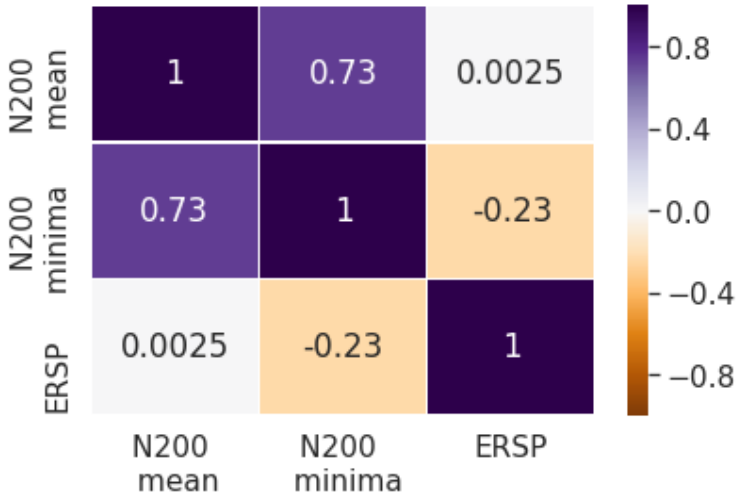


Figure 5.3: Correlation among the selected features

From this waveform, we extract three features to capture the user preference information for predictive analysis:

- N200 mean: The mean amplitude of the waveform is computed in the time interval of 200ms to 300ms (Figure 5.2).
- N200 minima: We also consider the minimum amplitude of the N200 interval as an additional feature.
- Event-Related Spectral Power (ERSP): The power spectral density of the waveform is calculated in the time interval of 100ms to 400ms in the beta frequency range i.e. 13 to 26 Hz (Figure 5.2). This PSD is calculated relative to the pre-stimulus baseline of 500ms.

We compute the Pearson correlation coefficient to explore the relationship between *N200* and *ERSP* features. We obtained a correlation coefficient of 0.0025 for *N200 mean* and *ERSP* indicating that the features are uncorrelated (p-value $= 0.0237 < 0.05$). As shown in Figure 5.3, *N200* and *ERSP* features are uncorrelated to each other. *N200 mean* and *N200 minima* have correlation coefficient of 0.73, as they present the similar time-domain aspect of ERPs (200ms - 300ms), i.e. average in the N200 duration is dependent on the minimum

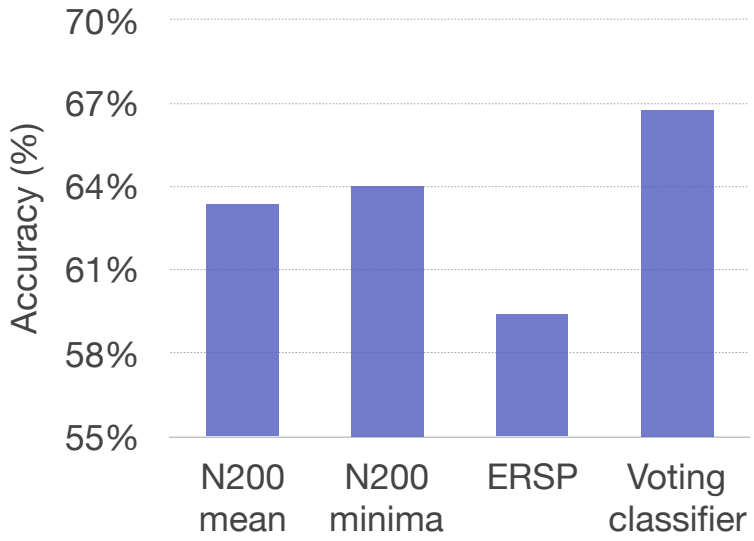


Figure 5.4: Pairwise classification accuracy for all products

during the duration to some extent. However, we consider *N200 minimum* separately, as in several cases, the effect of *N200 minima* is masked due to other baseline activities.

During our brute force trial and error analysis, we found that the combination of these features presents the most distinctive variability in the predictive analysis. The utility of *N200 mean* and *ERSP* in the beta band for predicting user preferences is also reported in [157, 156] and [268] respectively.

5.3.3 Establishing feasibility

In this subsection, we first validate the predictive capabilities of the selected EEG features through a pairwise choice classification task. Simply, we use the features as mentioned above in the EEG signals to determine pairwise preference with two objects at a time. We thus establish the feasibility of rank-ordering the objects using the pairwise results.

The task of pairwise choice classification involves mapping the neural measurement orderings to the preference amongt the consumer products. Thus, each neural feature (i.e., *N200 mean*, *minima* and *ERSP*) is independently used to predict the more preferred product in each product-pair. Specifically, the products having higher *ERSP* or higher magnitude

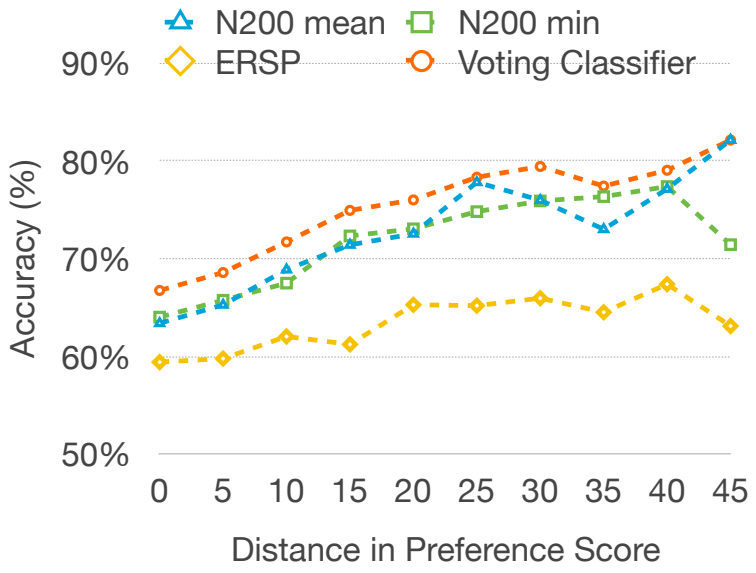


Figure 5.5: Accuracy for products with given distance in the preference scores

of *N200 mean* were found to have a lower preference, and the products with higher *N200 minima* had a higher preference. These comparison rules provide an accuracy of 63.38%, 64.01%, and 59.40% respectively for *N200 mean, N200 minima,* and *ERSP* (Figure 5.4). A voting classifier combining all three features performed with an accuracy of 66.7%. These metrics were computed on all pairs of products. As the difference between the preference scores between the two products compared increases, the accuracy increases as well (as can be seen in Figure 5.5). The maximum accuracy achieved is 82%. The accuracy of the decision classifier goes as high as 82% when the difference in relevance score is higher than 45.

Once pairwise preference can be determined, a naive ranking algorithm can be designed based on the relative ordering of one of the neural features. However, combining all of the three features is not as trivial as designing the decision classifier for the pairwise classification task. In addition, a fixed-comparison rule-based ranking algorithm will be oblivious to the individual differences (e.g., users with higher *ERSP* variations in comparison to *N200*), and hence, will not be able to generalize over a large set of users. We address these issues

in the next section by presenting the *Cerebro* solution.

The pairwise classification results presented above validates and establishes the predictive capability of the chosen features for preferences. A ranking scheme can be easily defined based on the ordering of neural features. However, it presents two challenges, (i) how to combine the neural features for ranking, (ii) how to determine which features work better for which subjects (i.e. some subjects have significant predictive capabilities in ERSP as compared to N200). We explore the learning algorithms in the next subsection to tackle these challenges.

5.4 The *Cerebro* Solution

Having established the feasibility of object ranking based on an EEG wearable, in this section, we present *Cerebro*, a machine learning algorithm that can learn the specific nuances of the user's waveforms for preferences, and is thus capable of ranking objects accurately.

5.4.1 Ranking algorithm

The feature designing approach explained in section 5.3.2, provides us with the rank-ordered neural features. The central idea behind learning paradigms is to understand and identify the individual differences and stochasticity among the users in their preferences. Learning algorithms adapt to the user-specific characteristics through training samples in order to reliably predict and rank the new products.

As described in section 5.3, the processed data for user u and product i, is a vector of neural features ($X_{u,i}$) and the preference score ($y_{u,i}$). *N200 mean* and *N200 minima* are transformed using function, $f(x) = 10log(1+x^2)$, to express *N200* features on the same scale as of *ERSP*. We build on the pairwise transformation ideas of learning to rank [269], and transform our dataset for each subject as,

$$\{X^{'}_{u,k}, y^{'}_{u,k}\} = \{X_{u,i} - X_{u,j}, sign(y_{u,i} - y_{u,j})\},\ i \neq j \tag{5.1}$$

i.e., for each product-pair, we use the relative differences in neural features, as our trans-

formed set of features. The labels are also transformed to +1 or -1 indicating if the i^{th} product was preferred more or less. This pairwise transformation enables the prediction of the relative order of products (which is critical in ranking) rather than the pointwise approach, which approximates the preference scores using neural features. The pairwise approach also helps in data augmentation as it creates $N*(N-1)$ samples from N samples. Here, X_i represents the vector of 3 neural features for i^{th} product described in the section 5.3.2, and y_i is the relevance score for the t^{th} product.

Based on the results in section 5.3.3, the relative order of the products is assumed to be linear with the given neural features. Hence, we fit a linear regression model[3] on the transformed set of features to predict the products with higher preferences. The regression model outputs a scalar value, which if positive, can be interpreted as the i^{th} product is preferable (or vice-a-versa, if negative). The linear model parameter β, on a conceptual level models the individual differences in terms of the importance of each feature for comprehending the user preferences. In the loss function of linear regression $L(\beta)$, we add a linear combination of L1 and L2 penalties for regularization in order to achieve a robust prediction.

$$L(\beta) = \frac{1}{|K|}\sum_{k}^{|K|} ||y_k^{'} - X_k^{'}\beta||^2 + \lambda_1||\beta||_1 + \lambda_2||\beta||^2 \tag{5.2}$$

L2 penalty (also known as Ridge regression) regulates the magnitude of the parameter β to tackle the over-fitting issue. L1 penalty (also known as Lasso regression) shrinks the coefficients of less important features to zero, thus, acts as a feature selection step. The optimal β^* is learned by minimizing the overall loss function eq. (5.2) over the training samples, $\beta^* = \arg\min L(\beta)$.

We learn a unique and optimal β_u^* for each subject u. Now, for user u, given the neural measure of a new product p (i.e., $X_{u,p}^{'}$), the preference score can be calculated by projecting the neural feature vector onto β_u^* i.e., $\frac{X_{u,p}^{'}.\beta_u^*}{||\beta_u^*||}$. The predicted preference scores are then compared to rank order the products.

[3]A classification model (e.g. RankSVM) is also an appropriate alternative.

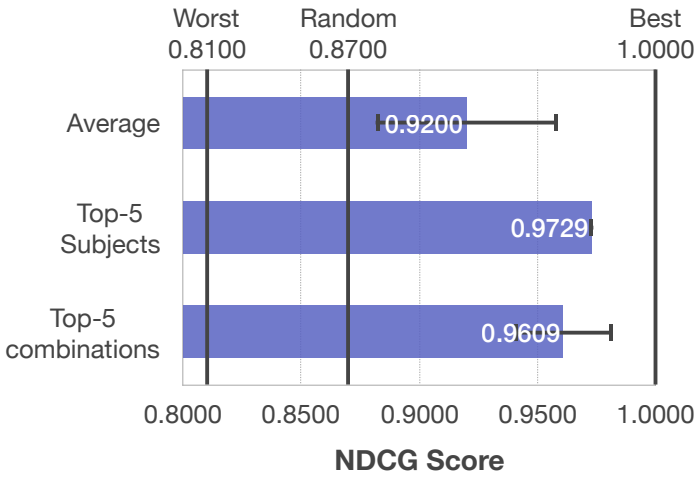


Figure 5.6: Ranking score: NDCG

5.4.2 Evaluation

Methodology: The ElasticNet [270] model was used to combine the L1 and L2 penalties in the linear regression model. λ_1 and λ_2 were set to 0.5. For each subject, we train the algorithm with 7 products, providing 42 training samples with pairwise transformation for the linear regression model. The algorithm was evaluated on the remaining 3 products by comparing the predicted ranking with the user-specified rankings. A total of 120 different training-testing sets are possible, hence, we present the performance metrics averaged over all the possible combinations.

Metrics: To evaluate the performance of *Cerebro*, we use two metrics, namely (i) MHD Score (Mean Hamming Distance), and (ii) NDCG Score (Normalized Discounted Cumulative Gain).

- **MHD Score:** MHD score computes the mean hamming distance between the predicted rank and the ground truth. For e.g., if a ground truth rank of y_{rank} =[1,2,3] is predicted as $\hat{y}_{rank}$ = [3, 1, 2], the MHD score would be 1.33. For 3 elements, the best, the worst, and random change MHD would be 0, 1.33 and 0.87 respectively.

- **NDCG Score:** It measures the ranking quality by accounting for the preference of

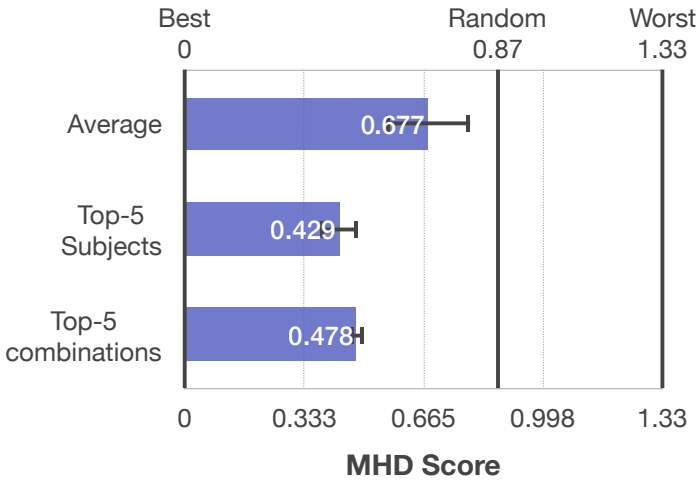


Figure 5.7: Ranking score: MHD

the products ranked. This metric is highly used in information retrieval and web search, as it gives a higher preference to the preferred products. NDCG is computed by normalizing the DCG score (Discounted Cumulative Gain), $\sum_{i=1}^{N} \frac{rel_i}{log_2(i+1)}$, with ideal DCG score. Here, rel_i represents the preference of the i^{th} product. An ideal DCG score would be the DCG score of products when ranked according to their preferences. For e.g., if the products with preference scores of y_{rel} =[30,20,10] are ranked as $\hat{y}_{rank}$ = [3, 1, 2], the DCG score would be 41.31, with an ideal DCG of 47.61, giving NDCG as 0.867. An NDCG score of 1.0 is ideal. For the preference scores in our dataset, a random chance NDCG is 0.87.

Performance: Figure 5.6 and 5.7 show the ranking performance on NDCG and MHD scores respectively. On average (14 subjects, 120 training combinations), the ranking algorithm performs with an NDCG score of 0.92 ($\pm$ 0.11) and an MHD score of 0.67 ($\pm$ 0.03). The considerable standard deviation in the ranking performance is due to the high variability of ranking performance across subjects and training combinations. Hence, we also evaluate the performance of top-5 subjects and top-5 training combinations. For top-5 subjects, the performance jumps to 0.973 ($\pm$ 0.0007) NDCG, and 0.429 ($\pm$ 0.052) MHD. Similarly, for top-5 training set combinations, we achieve 0.961 ($\pm$ 0.02) and 0.477 ($\pm$

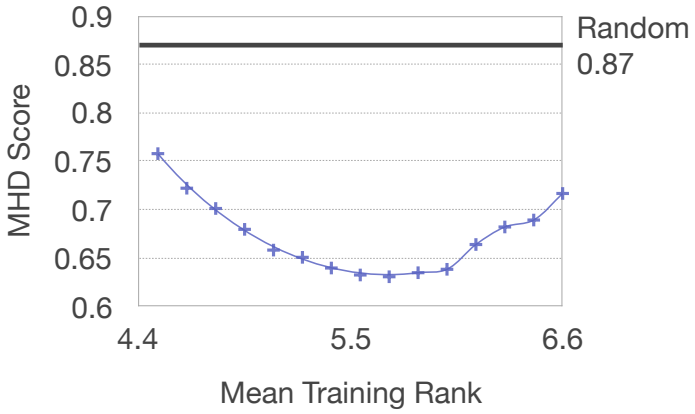


Figure 5.8: MHD Score on different training combinations

0.02) respectively.

5.4.3 Determination of confidence in ranking

Note that *Cerebro* requires user-training to understand and subsequently predict user preferences. One of the key questions that arise is the following - when is the algorithm trained enough such that it can start recommending objects according to the user preferences (i.e. when the neural signal based estimated preferences are actionable in real-world deployment)?

From the discussions thus far, we can observe that the performance of the ranking algorithm depends on the subjects and the set of product combinations chosen for training the algorithm. In this subsection, we explore whether it is feasible for the algorithm to self-determine if it has encountered the right set of products to be effectively trained. If such self-determination is feasible, the algorithm can begin predicting ranks for new objects only when it is sufficiently confident of its training.

Figure 5.8 shows the average MHD score (over all 14 users) with respect to the mean rank of the 7 products used in the training. With a larger spread of product ranking in the training set (mean training rank close to 5.5), it performs significantly better than with training products that are heavily biased towards top (or bottom) ranks. If the top 7 products

Table 5.1: Confidence in training

	Top-10	Worst-10
Training MHD	0.731	0.199
Testing MHD	0.476	0.876

are considered for training, it performs 20.1% worse than a uniform spread of training products (with a mean ranking of 5).

For practical self-determination of its confidence, the confidence measure should be solely based on the set of training products encountered thus far. In our dataset, we find that the combinations performing comparatively poor (in terms of training score of MHD or NDCG), tend to perform highly accurate on the testing data. A possible reason for this trend could be that the algorithm is exposed to the data with more variations, hence the training fit is reasonable (no overfitting), but more generalized to the unseen data.

Table 5.1 presents the MHD score of training combinations which has top-10 and worst-10 training accuracy. These results, while preliminary, shed light on an approach to predict confidence in performance for unseen products. When the training accuracy is tracked over time (with more number of products), the training and testing accuracy converge, indicating confidence. Our analytical approach is limited because of the small size (10 products) of the dataset. Another ideal approach to obtain confidence is through cross-validation [271]. The verification of this methodology is left for future work.

5.4.4 System architecture

In this subsection, we describe the system design allowing *Cerebro* to understand user preferences. There are three main components of the system architecture, namely, (i) *wearable device*, (ii) *mobile software*, and (iii) *cloud server*. The *wearable device* detects EEG signals and ships the digitized signals to the user's mobile device through a wireless link (Figure 5.9). The *mobile software* running on the user's mobile device processes the raw signals and extract neural features related to the user preferences as described in section 3.1 and 3.2. The computed features are sent to the *cloud server* which executes the *Cerebro*

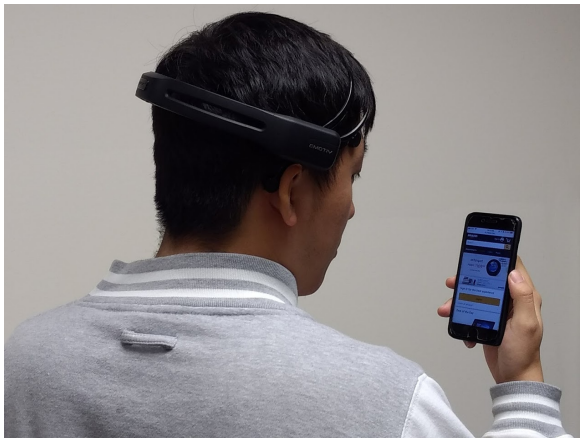


Figure 5.9: Use-case setup

algorithm to understand the user preferences and thus, ranks the objects. Finally, the analytical summary of the user preferences and ranking is sent to the concerned application server (e.g. Amazon personalization engine).

5.5 Summary

This work considers the potential of tracking neurobiological changes through wearable EEG headsets to understand user preferences. We study the detection and interpretation of user brainwaves to rank a given set of objects based on user preferences. We present *Cerebro*, a machine learning algorithm to enable objects ranking merely through the neural data based on user preferences. The performance of *Cerebro* is attractive, with an NDCG score of 0.92.

CHAPTER 6

ON USING BRAINWAVES AS IMPLICIT HUMAN FEEDBACK IN REINFORCEMENT LEARNING

Reinforcement learning (RL) is a class of approaches where an agent learns what action to perform for a given situation so as to maximize a numerical cumulative reward signal. RL is especially suited to uncharted territories where prior examples of correct actions are not readily available to the problem at hand. The agent is thus left to interact with the environment and learn from its own experience. The use of a reward signal to formalize the idea of a goal is one of the most distinctive features of reinforcement learning. While the notion of a simple reward signal has the advantages of being flexible and widely applicable, there still remains the challenge of defining an effective reward function in the first place. Engineering such a reward function can at times be non-trivial or noisy even when designed (for example, learning to backflip for a bipedal robot). In such scenarios, the RL algorithms might need to be supplemented with other strategies such as learning through demonstrations by an intelligent agent or human feedback. Methods like inverse RL (or learning through demonstrations), explicit human feedback (through labels, ratings, etc.) could reduce the search space or supplement the rewards, making the algorithm train more efficiently [272]. Human assisted machine learning, when combined with the need for RL to have access to rich reward functions, raises some significant challenges. This includes the conflict between the need to increase the richness of the reward function, while minimizing the burden placed on the human to generate the rewards.

In this work, we explore an interesting solution paradigm that allows humans to assist machine learning algorithms to substantially increase the richness of the reward functions, while not severely burdening the human-in-the-loop. Specifically, we study the use of electroencephalogram (EEG) based brain waves of the human-in-the-loop to generate the reward functions that can then be used by the machine learning algorithms. Such a model

benefits from the natural rich activity of a powerful sensor (the human brain), but at the same time does not burden the human since the feedback being relied upon is intrinsically generated. This paradigm is inspired by a high-level error-processing system in humans that generates error-related potential (ErrP). As such, while a human naturally monitors the performance of an agent, the erroneous behavior of the agent can be recognized intrinsically by the ErrP in the human EEG signals which we can build into the reward function of the RL algorithm of the machine to improve its intelligence.

This broad paradigm of using implicit feedback through brainwaves is broadly applicable to any application where a human can observe the agent in action and hence generate the intrinsic reactions. However, in order to systematically study different aspects of the paradigm, we use computer games as proxies for real-life environments that agents might need to operate in. The use of games as proxies for real-life environments is an interesting strategy in itself, as it has some distinct advantages including a highly controllable and replicable environment that offers clear control knobs that together can accelerate the pace of investigation and discovery.

In order to systematically study and design the practical framework to allow ErrP based implicit human feedback to accelerate RL algorithms, we provide our contributions in four research thrusts:

1. **Human experiments and systems research:** We develop custom-built game environments, experimental protocols and system framework to perform IRB approved human experiments. We identify and discuss key system issues with broader applicability, and conduct studies to quantitatively show the benefits of implicit feedback over manual human feedback.

2. **Error-potentials (ErrP) research:** We first provide experimental evidences for the ErrPs. We discuss state-of-the-approach to detect error-potentials and its major drawbacks. We propose *Trinity*, an algorithm to reliably detect error-potentials, and compare the performance over the collected datasets. Further, we provide additional

experimental analysis to gain an in-depth understanding of error-potentials.

3. **Integration with RL algorithm:** We discuss how we model the human feedback and the approaches to integrate the ErrP based feedback with RL algorithms namely action biasing, control sharing and reward shaping[1]. We evaluate the approaches in terms of their acceleration, and provide a sensitivity analysis of the reward shaping approach.

4. **Towards a practical solution:** We discuss methods to improve the practicality of the proposed system. Specifically, we explore two directions, (a) transfer learning in error-potentials, and (b) inverse RL approach for integrating the human feedback[1].

NOTE: The methodologies explained in this chapter to integrate error-potentials with a reinforcement learning algorithm, namely (a) reward shaping (section 6.5.1), and (b) learning from imperfect demonstrations (section 6.5.2), are contributions from our collaborators, Duo Xu and Dr. Faramarz Fekri. Their contributions are explained here in a very succinct manner for completion purposes. We thank our collaborators for their outstanding work and for providing us the permission to present their work in this thesis.

6.1 Background and Motivation

6.1.1 A primer on RL algorithms

Reinforcement Learning (RL) is a class of algorithms where an agent learns to make a good sequence of decisions (or act) in a given uncertain environment. The core idea in RL is to enable the agent to map situations to actions, in order to maximize a cumulative reward signal. The basic entities of RL are,

- **State(s_t):** State is the representation of the current situation or environment. For example, in a chess game, the state is the location of all the chess pieces on the board. A state can be modified when the agent performs an action.

[1]The research contributions are from our collaborators, Duo Xu and Dr. Faramarz Fekri. Their contributions are explained here in a very succinct manner for completion purposes. We thank our collaborators for their outstanding work and for providing us the permission to present their work in this thesis.

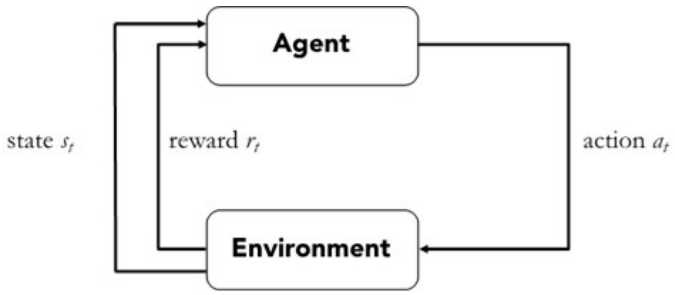


Figure 6.1: Reinforcement Learning (RL)

- **Action(a_t):** Actions are defined as the possible way the agent can act to modify the state in the environment. In the chess example, moving any piece within the set rules for the chess game is considered an action. Actions space refers to all the possible actions an agent can take within the given game environment.

- **Reward(r_t):** Rewards are the utility scores (scalar values) the agent receives upon performing the actions. The rewards in the chess game could be +1 for the win, and 0 for loss.

In this context, the agent interacts with the environment by taking an action a_t at time t, and the environment provides the next state s_t and reward r_t (Figure 6.1). The goal of the RL algorithm is to learn what actions to take in the given situation (or states) maximizing the total cumulative rewards. RL algorithms are suitable for situations where it is impractical to obtain examples of desired behavior that are representative of all the situations in which the agent must act.

Optimal policy in RL

As discussed before, the goal of the RL algorithm is to learn to map the situations to actions in an optimal manner (i.e., maximizing the total cumulative rewards). The mapping of states to actions defines the agent behavior, known as **policy(π)**.

$$\pi(a \mid s) = P(a_t = a \mid s_t = s) \tag{6.1}$$

The policy is known as deterministic if the mapping from states to actions is fixed. In case, the action mapping is probabilistic (e.g., in equation 6.1), the policy is known as stochastic.

Total return (G_t) are the discounted sum of rewards, i.e.,

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots \tag{6.2}$$

γ is known as the discount factor in the range of $[0,1]$. $\gamma = 0$ makes the agent myopic, i.e., focusing solely on the immediate rewards rather than the long-term goals. $\gamma = 1$ imparts the far sighted abilities to the RL agent. Moreover, the discount factors also helps to sum the infinite number of rewards in a tractable manner (i.e., in a finite manner).

For a given policy π, the goodness of a state can be defined using the state-value function $v_\pi(s)$ which provides an estimate of the expected return starting from the state s, and following policy π,

$$v_\pi(s) = E_\pi(G_t \mid s_t = s) \tag{6.3}$$

Action-value function (q_π) decouples the state and actions in the state-value function, and provides the expected return from state s, taking action a, and following policy π,

$$q_\pi(s,a) = E_\pi(G_t \mid s_t = s, a_t = a) \tag{6.4}$$

Using Bellman equation [273], the action-value function and state-value function can be written as,

$$v_\pi(s) = \sum_{a \in A} \pi(a \mid s) q_\pi(s,a) \tag{6.5}$$

Optimal state-value function ($v_*(s)$) is defined as the maximum value function over all policies,

$$v_*(s) = \max_\pi v_\pi(s) \tag{6.6}$$

Similarly, optimal action-value function (q_*) is defined as the maximum action-value function over all policies,

$$q_*(s,a) = \max_\pi q_\pi(s,a) \tag{6.7}$$

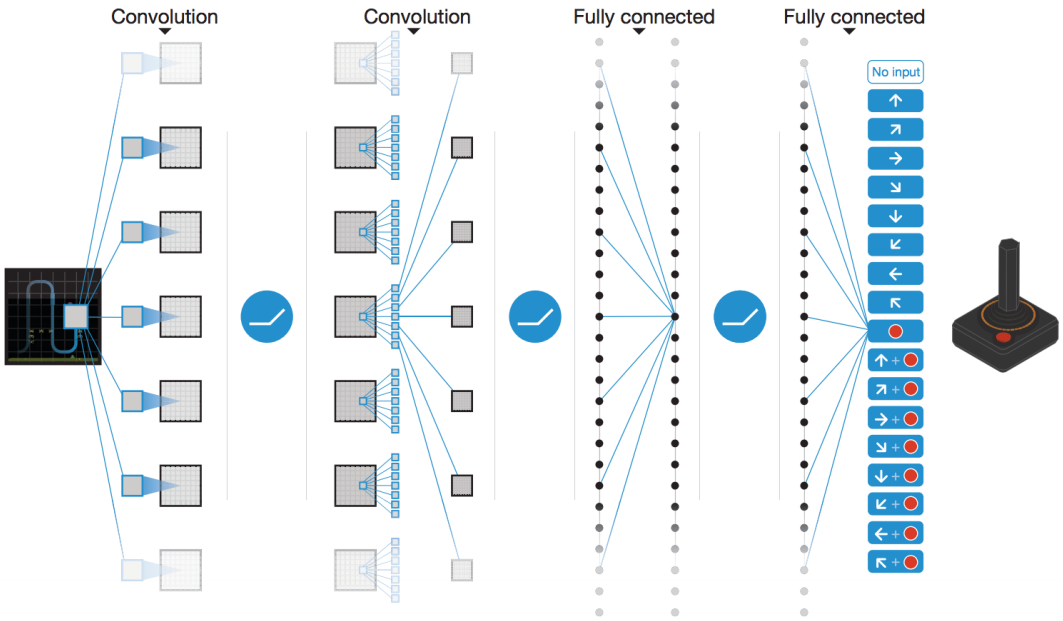


Figure 6.2: DQN architecture
Image taken from [274]

An optimal policy (π_*) can be derived by acting greedily according to the optimal action-value function (q_*),

$$\pi_*(a \mid s) = \begin{cases} 1 & \text{if } a = argmax_{a \in A} q_*(s,a) \\ 0 & \text{otherwise} \end{cases} \tag{6.8}$$

Thus, an optimal policy can be easily derived for any complex environment, if the optimal Q-function (i.e., q_*) is known.

Deep Q-Network (DQN)

Deep Q-Network or DQN [274] applies supervised learning-based function estimation techniques in reinforcement learning. It attempts to learn (or approximate) the optimal Q-value function (Q_*) with a deep neural network, i.e. $Q(s,a) = f(s,a,\theta_i)$, where θ_i represents the parameters or weights of the deep neural network. DQN takes input state (s) of 4

RGB snapshots of the game stacked together[2], and outputs the Q-value for all 19 actions. The architecture of the network consists of two convolutional layers [275] and one fully-connected layer (Figure 6.2). The first convolutional layers consist of 16 8x8 filters with Rectified Linear Unit (ReLU) activation. The second convolutional layer consists of 32 4x4 filters with ReLU activation. There are 256 hidden units present in the fully-connected layer.

DQN plays one step in the game, i.e., takes action a_t at state s_t, and receives reward r_{t+1} and new state s_{t+1}. DQN stores the tuple $(s_t, a_t, r_{t+1}, s_{t+1})$ in replay memory D. DQN makes use of experience replay to remove the correlations present in the sequential observations. While training, the labels (or tuples) are uniformly sampled from the replay memory, and used for training through Stochastic Gradient Descent (SGD). The loss function for the training is defined as the difference between estimated Q-value function, and target Q-value function based on the off-policy Temporal Difference (TD) control,

$$L_i(\theta_i) = E[r_t + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1}, \theta_i) - Q(s_t, a_t, \theta_i)] \tag{6.9}$$

However, in practice, the training is not stable, hence two separate networks are kept, one for playing the game (θ_i), and other for the target θ_i^-, hence, the loss function becomes,

$$L_i(\theta_i) = E[r_t + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1}, \theta_i^-) - Q(s_t, a_t, \theta_i)] \tag{6.10}$$

Every C steps, the parameters of the target network are updated with the Q-network. DQN also employs the use of epsilon-greedy strategy to allow the agent to reap the benefits of exploration and exploitation trade-off. In this strategy, at each time step t, an action is taken random with probability ε_t, and greedily according to the Q-network with probability $1 - \varepsilon_t$. With t, the ε_t is linearly reduced from 1.0 to 0.1 over 1M steps. DQN surpassed the performance of previous, and achieved human level performance on 57 Atari games from

[2]Input to the DQN is 84x84x4, where 84x84 is the downsampled and greyscaled snapshot (from 210x160x3 dimensional RGB snapshot), and 4 sequential frames (skipping 3 frames, i.e., considering every fourth frame) are stacked together

Atari 2600 suite.

6.1.2 Computer games and Atari benchmark

With the confluence of the considerable advancements in sensor technologies and processing power and the lowering of their respective costs, the use of machine learning solutions for cyber-physical systems (CPS) has indeed shown great promise. The broad paradigm of using implicit feedback through brainwaves to accelerate the learning of machine learning algorithms is broadly applicable to any application where a human can observe the agent in action and hence generate the intrinsic reactions. The machine learning solutions integrated with implicit human feedback (via brainwaves) are particularly useful for the monitoring, instrumenting, and optimization of complex CPS. One example of a complex CPS is *Surgical Robots*. A robot in this context is equipped with sensors that can sense the target environment (human body), a control architecture that processes the sensory data and generates actions, and end effectors or actuators that help the robot perform the actions. The key learning problem in this application is the mapping function from the perception of the environment to an action that needs to be performed in order to reduce the total cost incurred. This learning can be facilitated by one of several different approaches: the robot could learn on its own by evaluating the appropriateness of its own actions to reach particular target states; the robot could learn by observing human surgeons in action; or the human surgeon could also intervene and guide through the observation of the robot actions [276].

However, to systematically study different aspects of the paradigm, we use computer games as proxies for real-life environments that agents might need to operate in. The use of games as proxies for real-life environments is an interesting strategy in itself, as it has some distinct advantages including a highly controllable and replicable environment that offers clear control knobs that together can accelerate the pace of investigation and discovery. Consider a simple CPS example of a robotic vacuum cleaner that can detect when spills and messes occur inside a home, find its way to the location of the spill, and clean it up.

There are multiple learning problems embedded within this simple CPS example. The robot has to learn how to detect that a spill has occurred inside that specific home, learn its way around the home to get to the spill and learn how to clean up a spill based on what has been spilled. Consider specifically the navigation problem. Several RL algorithms can help the robot learn its way around the home to a specific goal [277, 278, 279]. However, these algorithms all require an external reward function from an oracle or an external system such as a camera rig that can provide distance-based rewards to the robot when it is navigating. While these assumptions are highly constraining in themselves, the latency required for the robot to learn its environment within its home is non-trivial and likely reoccurring given that obstacles will require re-learning by the robot. In this setting, consider a human-in-the-loop observer inside the home who is outfitted with a BCI cap to help with the robot's RL. As the robot is navigating its way through the home toward the spill, the passive human observer will naturally react to an observed subset of the robot's moves. This intrinsic reaction can be captured through the ErrP of the EEG signals captured by the BCI cap, and fed as a reward function back to the robot's RL algorithm.

Hence, using a game as a proxy for a real-life environment is beneficial in the context of human-assisted RL algorithms. Games are fertile ground for the definition, understanding, and improvement of RL algorithms in low overhead and speedy fashion. Games have now evolved to help understand the world around us and make optimal strategies to tackle various difficult and high-risk real-world situations. For example, Foldit is an online puzzle video game about protein folding. The users of the game helped to solve the structure of a protein-sniping enzyme critical for the reproduction of the AIDS virus. This was a problem that had stumped scientists for over a decade, and it took the game users three weeks to generate the insights that went into solving the problem. A curious planet with four stars was discovered through another game, Planet Hunter, along with the discovery of 40 other planets with the potential of having life-forms [280]. Motivated by these studies, we use games as environments for gathering ErrP data from humans to accelerate the off-the-shelf

RL algorithms.

Atari 2600 benchmark

Atari 2600 is a second-generation gaming console, massively popular for its more than 500 games including space invaders, Pong, Pacman, SeaQuest, etc. The games present a wide variety of challenges, encapsulating various real-world issues and thus requiring human-level control. The Atari console had a 1.19MHz CPU with 2-4kB cartridge ROM, and 1024 bit console RAM. The screen output resolution is 210 pixels in height, and 160 pixels wide with a 128 color palette. Total 19 actions are provided as an input to the games through the joystick including the NOOP (No Operation). More than 50 games from the Atari suite have become the standard benchmark to research and evaluate the reinforcement learning algorithms, to measure progress and successively build more intelligent agents, pertaining to their challenging and diverse set of tasks which could also be difficult for the human players.

OpenAI Gym

OpenAI Gym is a toolkit and software package developed by OpenAI for research and evaluation of reinforcement learning algorithms. It combines the benchmark collections (e.g., Atari games, robotic environments, etc.), and has a universal and accessible interface to interact with these environments. OpenAI module can be imported in Python to run and evaluate the agents driven by reinforcement learning algorithms. OpenAI Gym includes the Atari module emulating Atari games, built upon the Arcade Learning Environment (ALE). Along with the software library, OpenAI Gym also maintains a website to maintain the scoreboard for all the environments on RL algorithms submitted by the RL community.

6.1.3 Motivation

Reinforcement Learning (RL) algorithms have become an integral part of end-user applications, including autonomous systems (e.g., recommendation engines, self-driving cars, etc.), and robotics where the primary purpose of such systems is to understand and act in

unseen environments. Learning to make a good sequence of decisions in order to optimize rewards (a metric score, e.g., user satisfaction for personalized recommendations, revenue for advertisements, and scores for the computer games) forms the crux of RL algorithms. State-of-the-art algorithms (e.g., DQN [274], Rainbow [281]) perform with human-level control or superhuman performance, however, exhibit slow convergence rate [282]. This can be seen with the training time of DQN on simple Atari-games like Pong and Space Invaders. The training of DQN requires 1 million frames for Pong, and 10 million frames for Space Invaders. 10 million frames are equivalent to 46.27 hours of gameplay experience [3]. The slow convergence rate makes the RL algorithm inapplicable for real-world environments including robotic systems and autonomous vehicles.

One of the primary reasons for the slow convergence rate (or sample inefficiency) of RL algorithms is *reward sparsity*. Environments with *sparse rewards* or underspecified rewards, makes it extremely challenging for the agents to estimate the specific state-action pair leading to positive (or negative) rewards in a long sequence of actions. In this context, human feedback is shown to significantly improve the convergence of RL algorithms. Specifically, curriculum learning [283], auxiliary tasks [284], learning from experts [285], imitation learning [286], and inverse RL [287] are few approaches to solve the reward sparsity problem with the aid of external feedback from humans. In fact, human feedback based ML is quite ubiquitous around us (examples including Google Captcha training AI with human feedback, or ratings/reviews on Amazon, Netflix etc., for improved recommendation models). Several works identify the benefits of human feedback in the training of RL algorithms [288, 172]. However, in these works, human feedback is provided through keystrokes, touchscreen or using natural voice interfaces. The explicit requirement to take actions to communicate the feedback, burdens the human involved in the training loop of the RL algorithms, severely limits the applicability and scalability of such solutions. Thus, motivated from such limitations, in our work, we explore novel modalities to communicate

[3]human is playing the game rendered at 60 frames per second

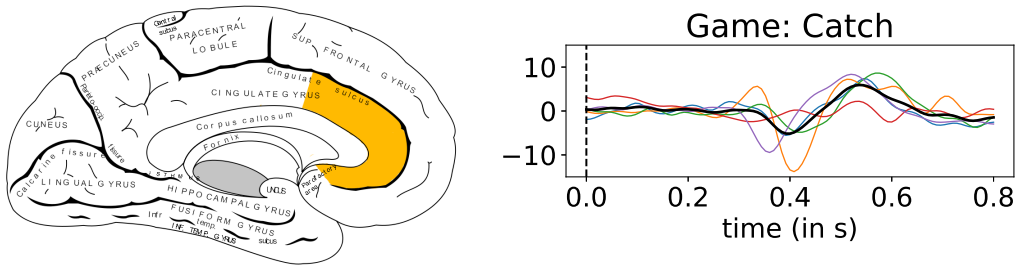


Figure 6.3: The left figure shows the Anterior Cingulate Cortex (ACC), the point of origin of the error-potential. The right figure shows the error-potentials over time-domain captured through a wearable EEG headset.
The colored lines represent the recordings from different subjects, and the solid black line represents the average over all the subjects.

or record human feedback to accelerate the training of RL algorithms.

6.1.4 A primer on error-related potentials

The electric potential elicited as a result of some specific activity in the brain typically reflecting a sensory, cognitive, or motor event, manifests as Event-Related Potentials (ERPs) in the human EEG. Numerous ERPs (e.g., N100, P300, N400, and P600) have been studied in detail, and causal relationships to external stimuli have been established with reasonable certainty in several cases. For example, the P600 ERP, which is a positive potential, is elicited when subjects detect a linguistic syntactic anomaly around 500 ms after the onset of the stimulus. Of particular interest to our application is the Error-Related Potential (ErrP), a negative potential elicited when a subject is presented with a stimulus that she perceives as an error [162] ((Figure 6.3 (right)). [289] analyzes the generality of the definition of "error" in the context of triggering ErrP and concludes that this brain potential is elicited for both errors of choice and errors of action. The authors also make the distinction that ErrPs are not triggered in response to the process of error correction in the subject's brain but as a result of the process of error detection. According to [290], the elicited ErrP is maximally negative at around 50 ms after the occurrence of the perceived error. The origin of ErrP has also been mapped to the Anterior Cingulate Cortex (ACC) in the brain [291] (Figure 6.3(left)). It has also been established that ErrP can be modulated by affective and

motivational factors [292]. In general, the amplitude of the elicited potential corresponds to the level of "startle response" that the error potentiated [292], meaning that it is possible to deduce the severity of the error algorithmically to some extent. Taking advantage of the nature of this brain wave, there have been numerous research attempts to exploit ErrP to aid in machine learning. For instance, in [293], the authors use the elicited ErrP as reward signals from a subject while a robot is solving a task. Further, the authors also demonstrated the feasibility of distinguishing different levels of errors from single-trial experiments. Extending further from this, [294] studies the feasibility of utilizing ErrP in online learning. There have also been attempts to infer and learn the strategy of a user using ErrPs [295].

Characteristics of error-related potentials

The signal is known to have two components, error-related positivity (P_E) and error-related negativity (N_E or ERN) [162]. In earlier studies, the N_E component was also found in correct trials [296]. Recent studies have found N_E to be more correlated to the errors and seem to reflect comparison processes [162]. As the source of N_E was found localized to the Anterior Cingulate Cortex (ACC), it is believed that the component reflects emotional and attention processes. However, P_E was found to be connected to the conscious error detection [297]. These components have different spatial distributions (N_E in fronto-central maximum, while P_E in centro-parietal maximum). In error-potentials, typically it is assumed that the error-potentials are the only signals that are time- and phase-locked to the stimulus. Hence, averaging multiple trials of these signals will provide insights into the template characteristic of the error-potentials.

6.2 System Overview and Data Collection

6.2.1 Game environments

We have first carefully designed and developed three discrete grid-based navigational games in OpenAI Gym *Atari* framework, namely Wobble, Catch, and Maze (Figure 6.4), summa-

[4]NOOP (No Operation) - the agent does not take any action at a particular time-step

Table 6.1: Description of the game environments

Game	Environment	Goal	Action space	Start/restart sequence
Maze	10x10 grid with an agent and a target	2D navigation to a fixed target in minimum number of steps	$\leftarrow$, $\downarrow$, $\uparrow$, $\rightarrow$	Maze is fixed for all instances.
Catch	10x10 grid with an egg and a basket, The egg falls one grid at each time instance	1D navigation by the basket to catch the egg at the right time.	NOOP[4], $\leftarrow$, $\rightarrow$	The egg starts at a random horizontal position from the top.
Wobble	1x20 grid with a cursor and a target	1D navigation to reach the target in a minimum number of steps	$\leftarrow$, $\rightarrow$	Cursor spawns at the center of the screen and target within 3 blocks of the cursor.

rized in Table 6.1, and explained below. We use the default Atari dimensions (i.e., 210x160 pixels with RGB color palette) with rendering at 60 frames per second. The games are designed in a way such that for any possible action that the agent takes, it is evident from the visual rendering of the game screen. The games are developed on Python and OpenAI Gym framework, with TCP/IP protocol to continuously transmit the state-action information from the game. The source codes of the games can be found in the public repository[5], and can be used with the OpenAI Gym module.

Wobble: We first designed Wobble, a simplistic 1-D cursor-target game, where the middle horizontal plane is divided into 20 discrete blocks. The cursor is shown with a big green square and the target is shown with small red (or blue) acquiring one block. At the beginning of the game, the cursor appears at the center of the screen, and the target appears no more than three blocks away from the cursor position. The action space for the agent is moving one step, either to the left or to the right. At each time step, the cursor can move one block in either direction. The game is finished when the cursor reaches the target. Once the game is finished, a new game is started with the cursor in place. The goal of the agent is to catch the target in minimum steps. Since the agent can take only left or right actions,

[5] `https://github.com/meagmohit/gym-maze`

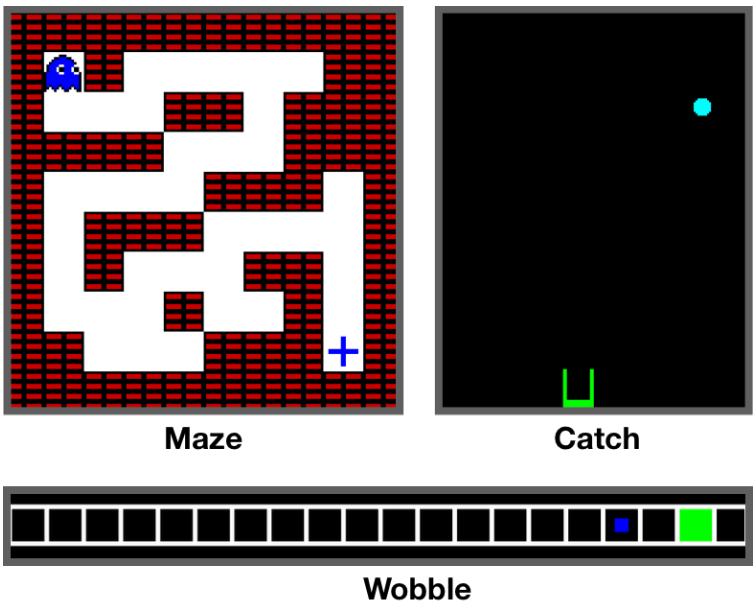


Figure 6.4: Game environments

it is easy for humans to see if the agent took correct or incorrect action.

Catch: We increased the complexity in our game environments by designing a new game, Catch, allowing the target to perform vertical movements in addition to the horizontal movements. The Catch game is a simplistic version of *Eggomania*[6] (Atari 2600 benchmark), where we display a single egg on the screen at a time. The Atari screen dimensions are divided into 10x10 grid space, where the "egg" and the "basket", both occupies one block. The action space of the agent consists of "NOOP" (no operation), "moving left" and "moving right". At the start of the game, the horizontal position of the egg is chosen randomly. At each time step, the egg falls one block in the vertical direction, and the goal of the agent is to catch the egg.

Maze: Our third game is Maze where we consider all four directional movements for the agent. Maze is a 2-D navigational game, where the agent has to reach a fixed target (shown with a *plus* symbol) in a minimum number of steps. The screen is divided into 10x10 square blocks. The action space consists of four directional movements. The maze architecture is kept fixed for this work. The only reward here is the result of the episode,

[6] https://en.wikipedia.org/wiki/Eggomania

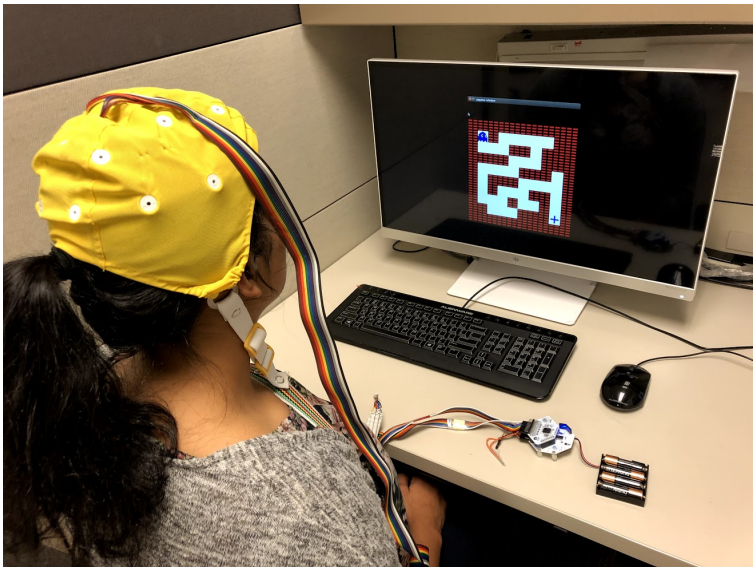


Figure 6.5: Experiment bench

i.e., win or lose. If an agent moves but hits a wall, a quick blinking of the agent is displayed, to render the action taken by the agent.

6.2.2 System overview and equipment

We designed and developed an experimental protocol, where a machine agent plays a computer game, while a human silently observes (and assesses) the actions taken by the machine agent (as shown in fig 6.5). These implicit human reactions are captured by placing raw electrodes on the scalp of the human brain in the form of EEG potentials. For capturing the raw analog brainwaves, we used the BIOPAC electrode cap (BIOPAC CAP-100C) with 16 EEG electrodes. The sixteen electrodes were Fp1, Fp2, Fpz, F7, F3, Fz, F4, F8, C3, Cz, C4, P3, Pz, P4, O1, and O2. Two additional Ag/AgCl electrodes were also clipped to the user's earlobes to provide the reference and additional noise correction mechanism. An electrode gel was injected to maintain the contact between the electrode and the scalp. The electrode cap was attached with the OpenBCI Cyton[7] platform, which was further connected to a desktop machine over the wireless channel. We used daisy module extension with OpenBCI Cyton to allow continuous sampling of brainwaves from 16 electrodes at

[7] http://openbci.com

Table 6.2: Error-Potentials: data collection

Game	Device	# users	# Stimulations
Maze	OpenBCI	12	≈ 7500
Catch	OpenBCI	8	≈ 5500
Wobble	OpenBCI	6	≈ 3700

125 Hz. ADS1299 designed by Texas Instruments, the heart of OpenBCI, converts the raw analog signals to digital samples. We used OpenViBE [96], a software platform developed in INRIA, France, to collect the digitized sampled brainwaves and synchronize them with the game status. OpenViBE continuously listens to the TCP port (for state-action pairs), and timestamps the EEG data in a synchronized manner. A detailed step-by-step procedure to conduct the human experiments is provided in section 6.9 along with the key system-level synchronization issues in section 6.10.

For the first phase of the experiments, we conducted more than 25 experiments with 6 subjects common (mean age 26.8 with a standard deviation of 1.92, 1 female) for all the three games (Table 6.2). We used standard recruitment and consent procedures for enrolling the human subjects in this study. For each subject-game pair, the experimental duration was less than 15 minutes. The agent took action every 1.5 seconds during the experiment. The Georgia Tech Institutional Review Board reviewed and approved all the research protocols for user data collection. This data is anonymized and stored for further analysis.

6.3 Benefits of ErrP based Implicit Feedback

6.3.1 Qualitative benefits of obtaining intrinsic feedback via error-potentials

Relying on error-potentials for obtaining intrinsic feedback provides two primary benefits - (a) generalized notion of error-detection instead of application specific, (b) strong signal-to-noise-ratio due to evolutionary significance.

Generalized notion of error-detection

Error-potentials are elicited when a user is presented with an incongruent (or erroneous) stimulus in a diverse set of tasks [298] implying that the error-processing system is generic (i.e., not specific to the task or sensory organ). Error-potentials are observed across a wide variety of input modality (e.g., audio [299], visual [300], somatosensory [301], etc.). This is in contrast to other elicited potentials in the brain which cater to the stimuli of a specific category. For instance, the P600, N300, P300, and N200 are elicited when a subject is presented with syntactic anomalies in sentences [302], semantically inconsistent word and picture pairs [303], interruption of a stimulus with another divergent stimulus [304], and detection of mismatch in a stimulus [305] respectively. Thus, the generalized mechanism for eliciting ErrPs is one of the characteristic advantages that it offers, unlike other brain-potentials specific to a stimulus or modality.

Evolutionary significance

Error-potentials in primates are well-founded and universal (exhibiting similar behaviors across individuals) as they have an evolutionary significance due to their importance in cognition, learning, and survival. Error-potentials enable the learning process via the administration of rewards and punishments in the Anterior Cingulate Cortex (ACC) [306]. In monkeys, error-potentials were generated in the anterior cingulate sulcus, when monkeys made errors in a simple response task. [307] found error-recognition units in monkeys' anterior cingulate sulcus that were activated when the animals received negative feedback in the form of the absence of an expected reward. Similarly, [308] found that when monkeys made errors in a simple response task, error-related potentials were generated in the anterior cingulate sulcus, thereby advocating that ErrPs link human and non-human primates based on error monitoring. The universality of ErrPs guarantees that it occurs naturally in humans and the evolutionary importance of ErrPs in learning points toward them being a foundational element in human cognition.

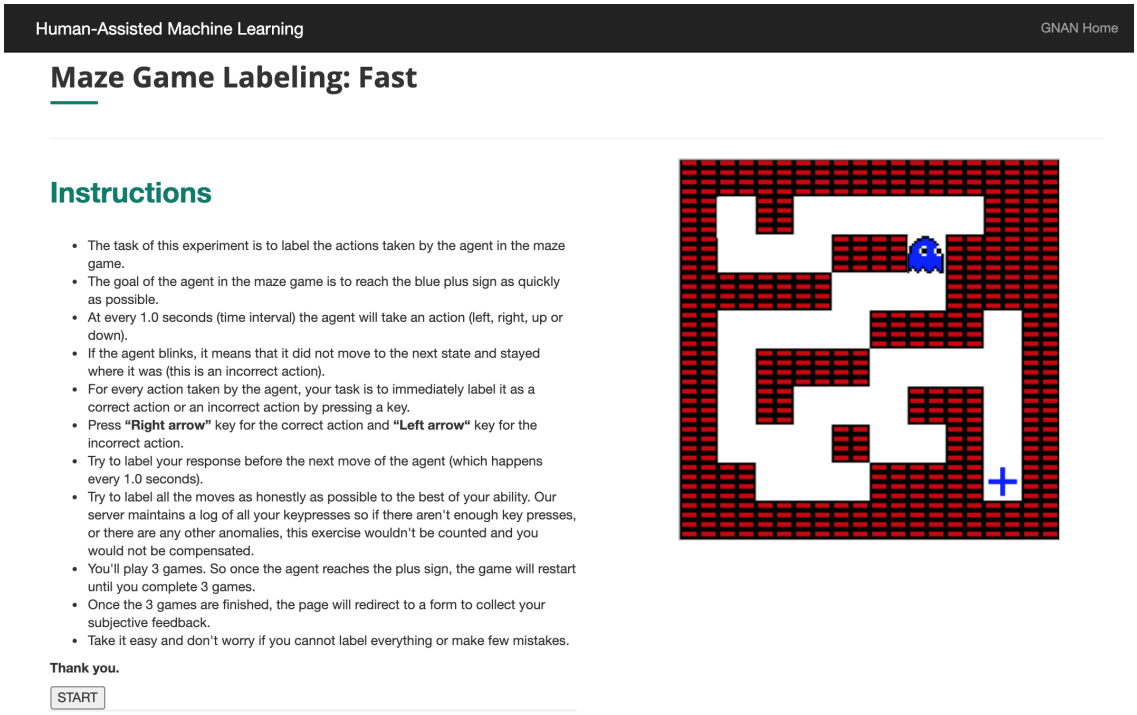


Figure 6.6: Experimental interface for evaluating manual feedback

6.3.2 Motivational study for using error-potentials over manual labeling

In this section, we describe the experimental study we conducted to quantitatively compare the benefits of intrinsic feedback (obtained via error-potentials) over manual feedback in terms of *accuracy*, *latency*, and *cognitive burden*.

Experimental methodology

We designed a web-based interface to conduct the experiments and to collect the data for quantitatively evaluating the manual feedback over intrinsic feedback. In the experiments, the subjects were presented with a Maze game screen (Figure 6.6) and were asked to label the actions taken by the computer agent in the game. The subjects were instructed to press the "left arrow" for incorrect action (taken by the computer agent) and the "right arrow" for the correct action. In total, 3 such instances of the game were designed, where each instance got progressively faster (to study the impact of time pressure on mental comfort and

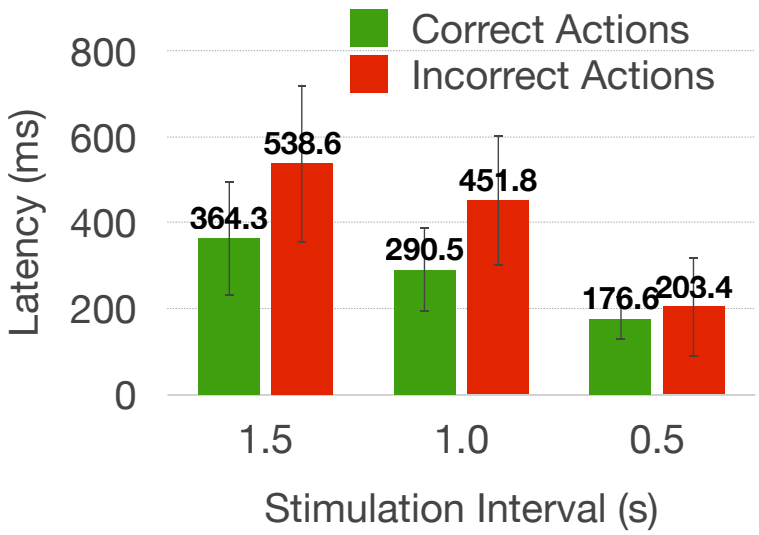


Figure 6.7: Latency in manual labeling

accuracy). The first instance had a time delay of 1.5 seconds between successive actions of the agent while the second and the third had a delay of 1.0 and 0.5 seconds respectively (we use these delay values as they lie around the latency value we have used in the EEG experiments and they also help us know the variation of manual labeling accuracy with respect to the latency). For each instance, every subject provided manual feedback over 3 trials (thus, 9 trials in total). The sequential order of the instances was randomized across users to remove any biases due to the ordering of the instances. In the maze game, the computer agent made the correct moves with a probability of 0.8. Upon the completion of all 3 trials for each instance of the game, the subjects were redirected to a Qualtrics survey where they were asked to provide subjective feedback (Table 6.3) about the experiment. Thus, there were 3 forms that each subject had to fill (one per instance). We used Amazon's Mechanical Turk to request anonymous workers to complete this task, and every worker was compensated 10 US cents upon successful completion of the task. The study was approved by Georgia Tech's Institutional Review Board.

Table 6.3: Qualtrics Questionnaire for 1.5s instance of the Maze game

Q1	Name
Q2	Email
Q3	Age
Q4	Rate the comfort scale of the experiment. Comfort scale refers to the cognitive load or the mental burden incurred upon you during the experiment (from 1 to 7)
Q5	Rate the comfort scale of the EEG experiment (If you participated in the EEG experiment earlier). Leave blank otherwise. (from 1 to 7)
Q6	Were you able to correctly mark *ALL* the actions taken by the agent within the time interval? (options: Yes, No, Not sure)
Q7	If no, at what time delay in between the agent actions, would you have been able to label it comfortably? [Options: (a) Greater than 1.5s but less than 3s (b) Greater than 3s but less than 4.5s (c) Greater than 4.5s but less than 6s (d) Greater than 6s (e) I was able to label it comfortably]
Q8	How do you think reducing the time interval from 1.5s to 1.0s would impact your labeling accuracy? [Options: (a) Considerably Increase (b) Slightly Increase (c) Stay the same (d) Slightly Decrease (e) Considerably Decrease]
Q9	How do you think reducing the time interval from 1.5s to 1.0s would impact your cognitive load or mental burden? [Options: (a) Considerably Increase (b) Slightly Increase (c) Stay the same (d) Slightly Decrease (e) Considerably Decrease]
Q10	Any feedback or comments for the experiment?

Results

We obtained a total of 281 responses for the conducted experiments. Specifically, we received 87, 91, and 103 unique user responses for the 1.5s, 1.0s, and 0.5s instances of the game respectively. On average, for the 1.5s instance of the game, we obtained a True Positive Rate (TPR) of 56.6% and 41.5% for correct and incorrect actions of the maze agent respectively. We also obtained a feedback latency of 376ms and 540ms for correct and incorrect actions of the maze agent respectively. It should be noted that correct and incorrect actions of the agent corresponding to the non-Errp and ErrP respectively, during EEG experiments. For the 1.0s instance, the TPR reduced to 49.8% and 38.8% (for correct and incorrect actions of the maze agent respectively), and further to 34.9% and 14.1% for the 0.5s instance. The significant decrease in TPR is intuitive as the subjects will not be able to respond accurately in an increased time pressure situation. The feedback latency (or reaction time) also decreased significantly, 288ms and 456ms (for correct and incorrect actions

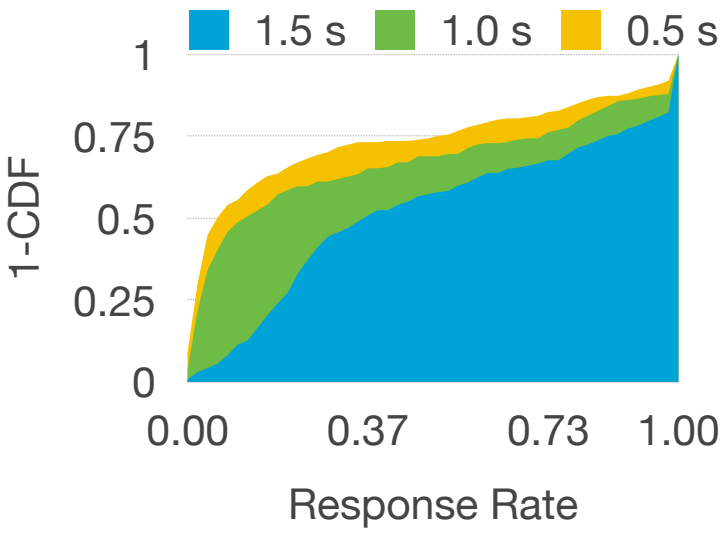


Figure 6.8: Histogram distribution of response rate for manual labeling

of the maze agent respectively) for the 1.0s instance, and further to 179ms and 207ms for the 0.5s instance of the game. Upon in-depth analysis of the raw data, we found that some participants were inert during the experiment, i.e., they were not actively participating in the experiment. To remove the biases due to such users, we decided to remove such inert participants for further analysis.

The response rate of the users for all three instances of the game is present in Figure 6.8. Here, we can see that the response rate for the slowest version of the game (1.5s instance) is highest as compared to the faster versions of the game. For each experiment, we removed the users who have less than 50% response rate. In other words, we removed the trials where participants failed to provide feedback for at least 50% of all the actions. This concluded in the removal of 22 users from the 1.5s instance of the game (25%), 28 users from the 1.0s instance of the game (31%), and 44 users from the 0.5s instance of the game (43%). After this filtering, for the 1.5s instance of the game, we obtained a true positive rate of 74.1% and 53.4% for correct and incorrect actions of the maze agent respectively. We also obtained a feedback latency of 364ms and 539ms for correct and incorrect actions of the maze agent respectively. For the 1.0s instance of the game, we obtained a true positive rate of 69.8% and 52.6% for correct and incorrect actions of the

Table 6.4: Accuracy and latency for manual labeling [Maze game]

Time Interval (s)	Subjects	TPR % (± std)		Latency (ms) ± std	
		Correct	Incorrect	Correct	Incorrect
1.5	87	74.06 (± 32.05)	53.37 (± 36.17)	364.26 (± 132.09)	538.59 (± 184.16)
1.0	91	69.79 (± 32.96)	52.56 (± 33.41)	290.50 (± 98.78)	451.79 (± 151.63)
0.5	103	56.36 (± 23.88)	21.60 (± 21.28)	176.56 (± 49.65)	203.44 (± 116.05)

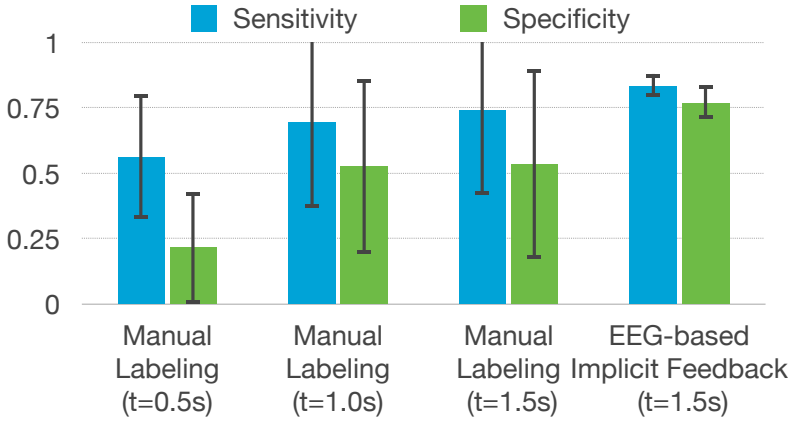


Figure 6.9: Comparison of manual labeling with implicit feedback (via EEG)

maze agent respectively. We also obtained a feedback latency of 290ms and 451ms for correct and incorrect actions of the maze agent respectively. For the 0.5s instance of the game, we obtained a true positive rate of 56.4% and 21.6% for correct and incorrect actions of the maze agent respectively. We also obtained a feedback latency of 177ms and 203ms for correct and incorrect actions of the maze agent respectively. The TPR and latency results are compared and summarized in Table 6.4, Figure 6.9, 6.7.

Insights

As we can see from the Table 6.4, Figure 6.9, 6.7, the accuracy values for correct and incorrect actions both decrease with decrease in the time interval. The labeling accuracy for correct actions is significantly more than that of incorrect actions. The accuracy for both, correct as well as incorrect actions, decreases as the time latency is decreased (thereby increasing time pressure). The highest accuracy for incorrect actions is about 53.4% (only marginally better than random labeling for 1.5s instance). In Figure 6.9, we have also provided the accuracy obtained for implicit feedback obtained (via error-potentials). The

implicit feedback based accuracy for correct actions is 83.61% (i.e., absence of error-potential), i.e.,12.88% improvement over manual labeling. Similarly, for incorrect actions, the implicit feedback accuracy (i.e. presence of error-potentials) is 77.02%, an improvement of 44.31% over manual labeling.

Based on the qualitative survey responses, on a scale of 1 to 7, the users gave the 1.5s instance of the game an average comfort rating of 5.4 which declined to 4.9 and 3.9 for the 1.0s instance and 0.5s instance respectively. On being asked if they were able to mark all actions correctly, 40% of the subjects answered in the affirmative in the 1.5s instance of the game, which declined to 26% and 14% in the 1.0s and the 0.5s instance of the game. Across the board, the majority of the participants reported that the ideal time interval for them to correctly label all actions of the agent would be between 1.5s and 3.0s or larger. 64% of the participants in the 1.5s instance of the game reported that reducing the time interval of the game to 1.0s would decrease their labeling accuracy, and 69% of the participants reported that it would increase their mental burden. 52% of the participants in the 1.0s instance of the game reported that reducing the time interval of the game to 0.5s would decrease their labeling accuracy, and 60% of the participants reported that it would increase their mental burden. In contrast, 64% of the participants in the 1.0s instance of the game reported that increasing the time interval from 1.0s to 1.5s would increase their labeling accuracy and decrease their mental burden. 49% of the participants in the 0.5s instance of the game reported that reducing the time interval of the game further would decrease their labeling accuracy, and 53% of the participants reported that it would increase their mental burden. To summarize, the users felt increasing discomfort and cognitive burden as the time latency reduced from 1.5s to 1.0s and further to 0.5s. They also reported that the optimal time latency for comfortable manual labeling would be between 1.5s and 3.0s. This was also evident from the fact that more than 60% of the participants anticipated a reduction in their accuracy if time latency was to be decreased from 1.5s.

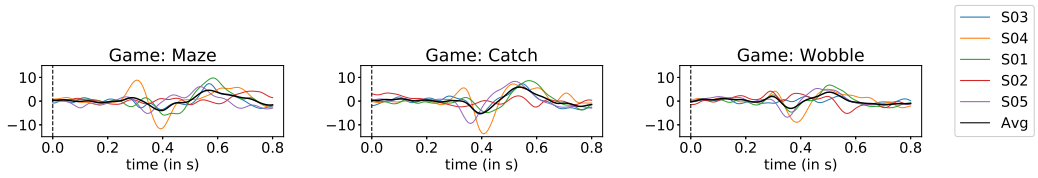


Figure 6.10: Manifestation of error-potentials in time-domain: Grand average potentials (error-minus-correct conditions) are shown for Maze, Catch and Wobble game environments.
Thick black line denotes the average over all the subjects.

6.4 Detection and Study of Error-Potentials

In this section, we discuss the algorithms to detect the presence of error-potentials directly from the captured brainwaves (i.e., EEG). We first validate that the observed neural correlates are error-potentials through various experimental and data analysis in section 6.11. Further, in Figure 6.10, we plot the grand average EEG waveforms across three environments (Maze, Catch, and Wobble), to visually validate the consistency of potentials for the five subjects. We can see that the shape of negativity and the peak latency is quite consistent (as per the literature) across the three game environments.

6.4.1 Baseline algorithm for detection of error-potentials

In order to obtain the implicit human feedback, we need to detect the presence or absence of ErrPs inside the EEG waveform. This requires training a model that can interpret the EEG signal of a human and classify it as an ErrP or non-ErrP robustly. EEG signals are inherently very noisy, and when combined with external factors like improper electrode placements, variance across users pose significant challenges in the reliable estimation of error-potentials.

We rely on the Riemannian Geometry framework for the classification of a human's intrinsic reaction [309]. This framework is state-of-the-art for detecting any event-related potentials, and provides two primary advantages over other classifiers [8]:

[8]The authors successfully applied the framework and won multiple Kaggle challenges. E.g., `https://www.kaggle.com/c/inria-bci-challenge`. Later, this framework was successfully adapted in many other error-potential decoding works [48].

- The estimation algorithm operates in signal space (rather than source space), and hence minimizes the distortions due to the electrode placements.
- The spatial filtering algorithm maximizes the signal to signal plus noise ratio (SSNR) to mitigate the interference and noise.

Algorithm 3: Riemannian Geometry based ErrP classification algorithm [166]

Input : raw EEG signals (X)

$X_f \leftarrow$ filtering(X, $freq\_band$, $filter\_order$) ;
$X_C \leftarrow$ covariance(X_f) ;
$X_D \leftarrow$ electrode_select (X_C, $nelec$) ;
$X_T \leftarrow$ tangent_space(X_D) ;
$X_N \leftarrow$ normalization(X_T, norm="l1") ;
$score \leftarrow$ elasticnet (X_N, λ_1, λ_2) ;
if $score > score_{th}$ **then return** True ;
else return False. ;

The algorithm parameters are explained in section 6.4.3

The principal idea in this approach is underpinned on the assumption that spatial distribution and power of the signal remain unaltered for a specific mental activity, which can be captured using the covariance matrix. Since the space of the covariance matrices is a subspace of Symmetric Positive Definite (SPD) matrices, it forms a differentiable Riemannian manifold. In this manifold, (i) the tangent space has an inner product that varies smoothly, and (ii) the distance between two points can be computed using Riemannian distance (or *geodesic*, δ_R) defined as,

$$\delta_R(C_1, C_2) = ||log(C_1^{-1}C_2)||_F = \left[\sum_{i=1}^{n} log^2(\lambda_i)\right]^{\frac{1}{2}} \tag{6.11}$$

Here, C_1 and C_2 represent the covariance matrices (corresponding to different data trials). $||.||_F$ represents Frobenius norm, and λ_i represents the i^{th} eigenvalue of $C_1^{-1}C_2$. One of the unique properties of this space is, $\delta_R(W^TC_1W, W^TC_2W) = \delta_R(C_1, C_2)$, for all invertible SPD W, implying that this space is invariant by projection (and hence less prone to noise and imperfect cap placements). The full algorithm is presented in Algorithm 3 and is explained below.

Algorithm Description: (Step 1) The first step is to bandpass filter the raw EEG data in a frequency range ($freq\_band$) of [0.5, 40] Hz, and epochs of 800ms duration were extracted relative to the pre-stimulus 200ms baseline. The epochs were then spatially filtered with "xDAWN Spatial Filter" [310, 309, 167]) to improve the signal to signal plus noise ratio (SSNR), where $filter\_order$ corresponds to the Xdawn components used to decompose the data for each event type. Such responses ($\mathbf{P}$) are obtained by taking the grand averages of training samples in each class (i.e. "non-ErrP" and "ErrP"), and a super trial ($\tilde{\mathbf{X}}_i$) is built by concatenating the class trials with their prototyped class response.

$$\mathbf{P} = \frac{1}{N}\sum_{i}^{N}\mathbf{X}_i, \quad \tilde{\mathbf{X}}_i = \begin{bmatrix} \mathbf{P} \\ \\ \mathbf{X}_i \end{bmatrix} \tag{6.12}$$

(Step 2) A covariance matrix is computed using the super trials ($\tilde{\mathbf{X}}_i$) accounting for the spatial distribution of the signal power. (Step 3) To overcome the curse of dimensionality, the covariance matrix is reduced by applying a channel selection algorithm. The procedure consists of a backward elimination with the Riemannian distance between the Riemannian Geometric mean of the covariances of each class as the criterion [311]. (Step 3) As the raw input signal is high-dimensional, the spatially filtered signals are reduced to fewer relevant channels ($nelec$) using a backward elimination principle based on the Riemannian distance between spatial covariance matrices as the selection criterion [311]. (Step 4) The reduced covariance matrix is projected into the tangent space, allowing to manipulate features in the Euclidean space [312, 166]. (Step 5, 6) Finally, the features in the tangent space (X_T) are normalized using the L1 norm and subjected to a linear regression model with L1 (λ_1) and L2 (λ_2) penalties. If the output of linear regression crosses the preset threshold ($score\_th$), the signal is labeled as an ErrP. $score\_th$ is set offline through maximizing accuracy over training samples.

6.4.2 *Trinity* algorithm

The baseline algorithm relies only on the spatial distribution of the scalp potentials (through the estimation of the covariance matrix) to classify the error-potentials. Despite the state-of-the-art performance of the algorithm, there is significant room for improvement. In practical situations, the error-potentials are not exactly time-locked, and manifest phase jitters due to the shift in user focus, synchronization issues (section 6.10), etc, resulting in reduced classification performance. Further, the distribution of power across time- and frequency-spectrum is known to provide additional information regarding the associated mental activity. In this section, we present our proposed algorithm, *Trinity* to supplement the spatial- domain features along with the time- and frequency- domain features and we efficiently combine the information across these three dimensions based on a soft-voting based ensemble approach (presented in Algorithm 4).

Algorithm Description: Pre-Processing: We use the bandpass filtering in [0.5, 15] Hz, and epoch extractions of 800ms relative to 200ms baseline. The signals were spatially filtered, projected to source space using "xDAWN spatial filtering", and were subjected to three pipelines independently.

Spatial-domain-based: For extracting the spatial features, we rely on the Riemannian Geometry framework proposed in [309, 167], and explained in the previous subsection. Instead of regression, we use a squared hinge loss [313] along with L1 and L2 penalties, and train with Stochastic Gradient Descent (SGD) [314]. In addition, we obtain the calibrated confidence scores (p_s) for spatial-domain based prediction based on [315, 316].

Frequency-domain features: (Step 7) A multi-taper spectral estimation method [317] within 400ms to 1000ms time window ($time_f$) after stimulus onset is used to compute the power densities in 1-15 Hz frequency interval ($freq_f$). (Step 8) The obtained power spectral values are converted to a logarithmic scale (dB). (Step 9) A linear-kernel based Support Vector Machine (SVM) with a small-margin hyperplane is used to classify the frequency-based features, and the confidence scores (p_f) are estimated using Platt scaling

[318, 319].

Time-domain features: (Step 8) The spatially filtered signals are divided into multiple buckets (*bucket_size*) of 50ms each. (Step 9) We compute the average amplitude of each bucket as the raw features representing time-domain variations in error-potentials. (Step 9-10) The mean amplitude-based features are normalized using the L2 norm, before feeding them to the linear SVM. Similar to the frequency-domain pipeline, we compute the probability estimations (p_t) representing the prediction confidence. The probability estimates were fed to the ensemble classifier.

Ensemble classification: We use a soft voting based ensemble classification to predict the "ErrP" or "non-ErrP" class. In this method, we average the classification probability i.e. p_t, p_f and p_s to compute the final estimation probability, p. To improve the overall detection performance of the system, we discard the low-confidence predictions. We define a parameter, probability threshold (p_{th}), to identify the low-confidence predictions. If the ensemble classifier prediction probability (i.e., p) lies between $[1 - p_{th}, p_{th}]$, we discard the corresponding samples.

6.4.3 Evaluation

We first validate the feasibility of decoding error-potentials using a 10-fold cross-validation scheme for each game relying on the Riemannian Geometry framework (state-of-the-art algorithm as explained in Algorithm 3). The code for the state-of-the-art algorithm was obtained from the public GitHub repository of the authors[9]. We used the IEEE BCI-NER challenge pipeline with the algorithm hyperparameters presented in Table 6.5. The algorithms are evaluated on the data collected for three environments, namely Maze, Catch, and Wobble (as explained in section 6.2.1). A separate classifier is used for each user and each game, i.e., algorithm learnable parameters are not shared across users and game, to demonstrate the feasibility of the detection of error-potentials.

In this scheme, we split the state-action pairs of a game into 10-folds for training and

[9]https://github.com/alexandrebarachant/bci-challenge-ner-2015

Algorithm 4: *Trinity*: Proposed algorithm for the classification of error-potentials

Input : raw EEG signals (X)

$X_f \leftarrow$ filtering(X, $freq\_band$, $filter\_order$) ;

/* Spatial Filtering */

$X_C^S \leftarrow$ covariance(X_f) ;

$X_D^S \leftarrow$ electrode_select (X_C^S, $nelec$) ;

$X_T^S \leftarrow$ tangent_space(X_D^S) ;

$X_N^S \leftarrow$ normalization(X_T^S, norm="l1") ;

$p_s \leftarrow$ linear_classification(X_N^S, λ_1, λ_2) ;

/* Frequency-domain */

$X_T^F \leftarrow$ multitaper_PSD(X_f, $time_f$, $freq_f$);

$X_N^F \leftarrow$ log_normalization(X_T^F) ;

$p_f \leftarrow$ svm(X_N^F) ;

/* Time-domain */

$X_B^T \leftarrow$ time_bucketing(X_f, $bucket\_size$);

$X_P^T \leftarrow$ average_power(X_B^T);

$X_N^T \leftarrow$ normalization(X_B^T, norm="l2") ;

$p_t \leftarrow$ svm(X_N^T) ;

/* Ensemble Learning */

$p \leftarrow$ soft_voting (p_s, p_f, p_t) ;

if $p > p_{th}$ **then return** True ;

else if $p < 1 - p_{th}$ **then return** False ;

else return None. ;

The algorithm parameters are explained in section 6.4.3

Table 6.5: Algorithm hyperparameters for the state-of-the-art algorithm for ErrP detection

Parameter	Value
Frequency Range	1-40 Hz
Frequency Filtering	Bandpass 4th order
Baseline Epoc window	100ms
Epoc window	1300ms
xDAWN Spatial Filters	4
Backtrack Electrodes ($nelec$)	8
ElasticNet: L1 Ratio	0.05
ElasticNet: α	0.02

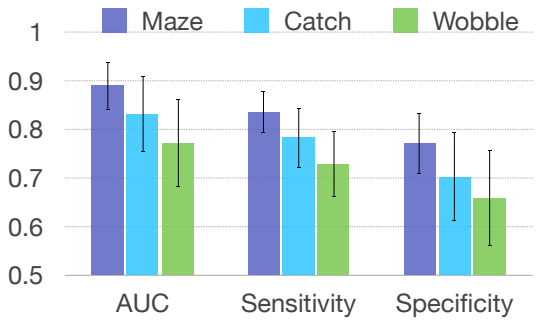


Figure 6.11: Feasibility of ErrP detection (state-of-the-art algorithm)

validation of the error-potential decoder. In Figure 6.11, we show the performance of three games in terms of Area Under Curve (AUC) score, sensitivity, and specificity averaged over 6 subjects. *Sensitivity* measures the true positive rate of a classification scheme. In our context, sensitivity refers to the percentage of time error-potentials are correctly classified, i.e., total correctly detected error-potentials out of given ground truth error-potentials. *Specificity* measures the true negative rate of a classification scheme. In our context, specificity refers to the percentage of time non-ErrPs (i.e., states where the agent took the correct actions) are correctly classified, i.e., total correctly detected non-ErrP out of given ground truth non-ErrP. *Area Under Curve (AUC)* computes the area under the receiver operating characteristic curve and provides a measure of the separability of the two classes. AUC does not rely on a particular value of the threshold, and hence provides insights into the goodness of the fit independent of the threshold value chosen. AUC score is 1 for an ideal classifier.

The Maze game has the highest AUC score (0.89 ± 0.05) followed by Catch (0.83 ± 0.08) and Wobble (0.77 ± 0.09). Sensitivity and specificity follow the same trend. We obtained a 0.83 ($\pm$ 0.04) score for sensitivity for the Maze game, and 0.78 ($\pm$ 0.06) and 0.73 ($\pm$ 0.07) for the Catch and Wobble game respectively. Similarly, sensitivity scores are obtained as 0.77 ($\pm$ 0.06), 0.70 ($\pm$ 0.09), 0.66 ($\pm$ 0.09) respectively for Maze, Catch and Wobble game respectively.

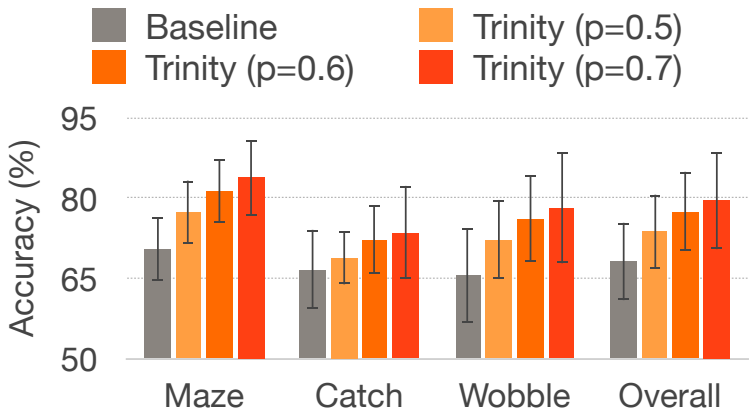


Figure 6.12: Accuracy

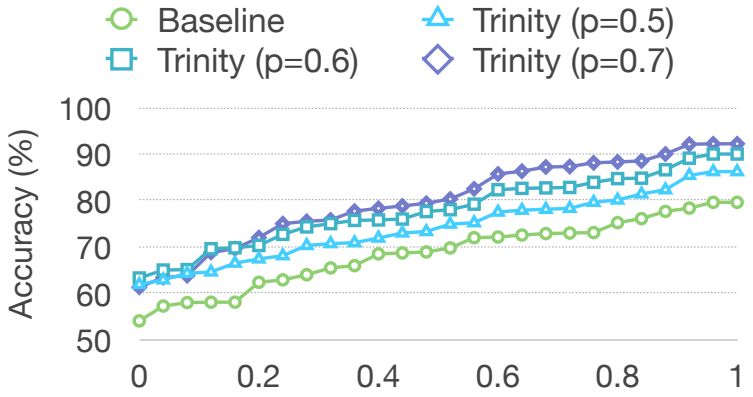


Figure 6.13: Accuracy CDF

Trinity performance

In this subsection, we evaluate the performance of the proposed error-potential decoding algorithm, *Trinity*, and compare it with the baseline algorithm. For the *Trinity* algorithm, we have set the $filter\_order$ to 4 (for xDAWN Spatial Filtering), and, λ_1 and λ_2 to 0.001 and 0.02, respectively. The proposed algorithm is evaluated over the probability threshold parameter p_{th}. The algorithms are evaluated on the data collected for three environments, namely Maze, Catch, and Wobble (as explained in section 6.2.1). The evaluation was performed using a 10-fold cross-validation scheme, and a separate classifier is used for each subject and each game. We also present the *overall* performance over all the subjects and the game environments in terms of *accuracy*, and *sample efficiency*. Accuracy presents the average accuracy of both classes (ErrP and non-ErrP) weighted equally. The diagonal

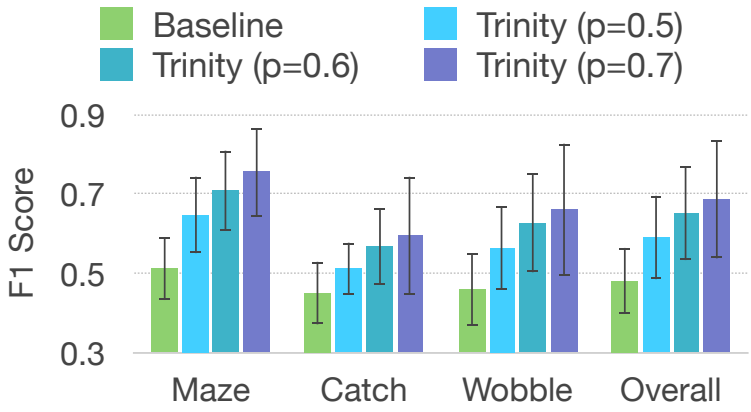


Figure 6.14: F1 Score

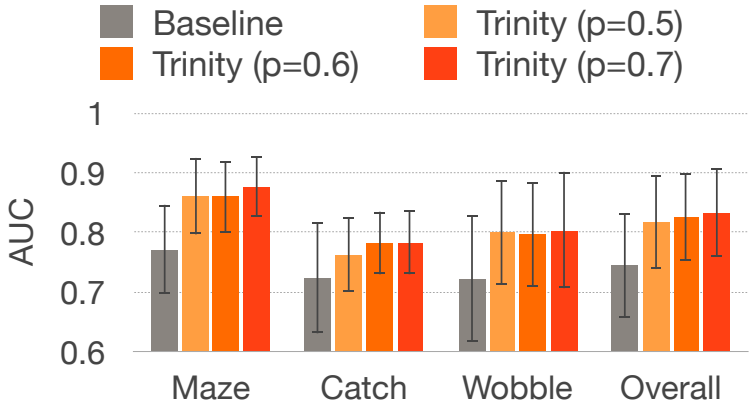


Figure 6.15: Area Under Curve (AUC)

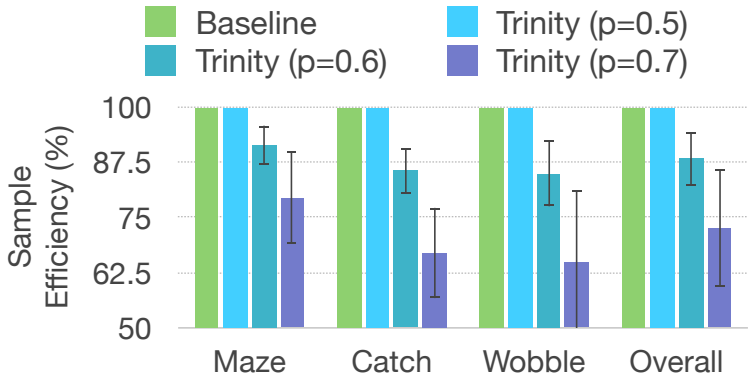


Figure 6.16: Sample efficiency

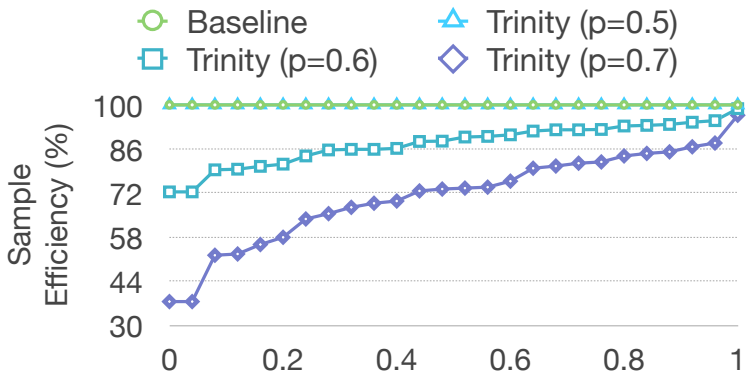


Figure 6.17: Sample efficiency CDF

elements of the confusion matrix are averaged to compute the average accuracy. Sample efficiency provides the percentage of data samples that can be confidently assigned to one class. Note that sample efficiency is 100% for the algorithms where none of the samples are dropped. As we increase the threshold to drop samples as per the measured confidence score, the sample efficiency reduces.

We present the overall detection accuracy of the proposed algorithm and compare it with the baseline in Figure 6.12. The proposed algorithm without discarding any samples (p_{th}=0.5) performs with an average accuracy of 73.71% ($\pm$ 6.81), an 8.11% improvement over the state-of-the-art. The accuracy is further boosted to 77.47% (13.6% improvement) and 79.51% (16.63% improvement) by increasing the p_{th} (dropping the low confidence samples) to 0.6 0.7 respectively. This improvement is achieved at the cost of sample efficiency of 88% ($\pm$6.01) and 72.3% ($\pm$13.33), for the p_{th} value of 0.5 and 0.6 respectively (as shown in Figure 6.16). Among all three games, the accuracy rate of the Maze game (77.28%) is higher pertaining to its simple and intuitive user interface.

Figure 6.13 presents the cumulative distribution of accuracy over a total of 25 recordings. It can be noted that for 50% of samples, the baseline algorithm performs over 70%, while the proposed algorithm (with $p_{th=0.5}$) performs over 80%. This trend is more clearly seen in Figure 6.13, where the cumulative distribution of the proposed algorithm with higher p_{th} lies over those with lower p_{th} and the baseline algorithm below all others. In

Table 6.6: Algorithm hyperparameters for the *Trinity* algorithm

Parameter	Value
Frequency Range	1-15 Hz
Frequency Filtering	Bandpass 4th order
Baseline Epoc window	200ms
Epoc window	800ms
xDAWN Spatial Filters	4
Backtrack Electrodes ($nelec$)	8
$[t_{min}, t_{max}]$	$[0.4, 1.0]$
$[f_{min}, f_{max}]$	$[1, 14]$
Frequency bins	16
Time buckets	50ms
ElasticNet Penalty: L1 Ratio	0.05
ElasticNet Penalty: α	0.02

Figure 6.17, we present the cumulative distribution of sample efficiency over all subjects. The baseline algorithm and proposed algorithm (with $p_{th} = 0.5$) perform with 100% sample efficiency since no sample is dropped. However, increasing the low-confidence threshold range, i.e., p_{th}, the sample efficiency reduces. For $p_{th} = 0.6$, the sample efficiency is above 85% for at least 75% of the users, making the algorithm practical and universal for subjects. With $p_{th} = 0.7$, the classifier performs with very high accuracy, with a sample efficiency of over 50% for more than 90% of the users. This simply translates to the fact that one out of two error-potential can be effectively labeled with this approach.

There is a clear increase in the average overall accuracy in comparison with the baseline algorithm (68% to 74%) with $p_{th} = 0.5$. From Figure 6.12, an increase is also observed in accuracy when samples with lower confidence are dropped (that is, as p_{th} is increased to 0.6 and 0.7). Using $p_{th} = 0.7$ it is seen that the overall average accuracy increase is over 11% and nearly 14% in Maze specifically. This trend is more clearly seen in Figure 6.13, where the CDF of the proposed algorithm with higher p_{th} lies over those with lower p_{th} and the baseline algorithm below all other curves. The performance improvement can also be observed from Figure 6.15 in the AUC scores where the overall average over three games is 74.4% for baseline algorithm and 83.2% with $p_{th} = 0.7$ for the proposed algorithm (9%

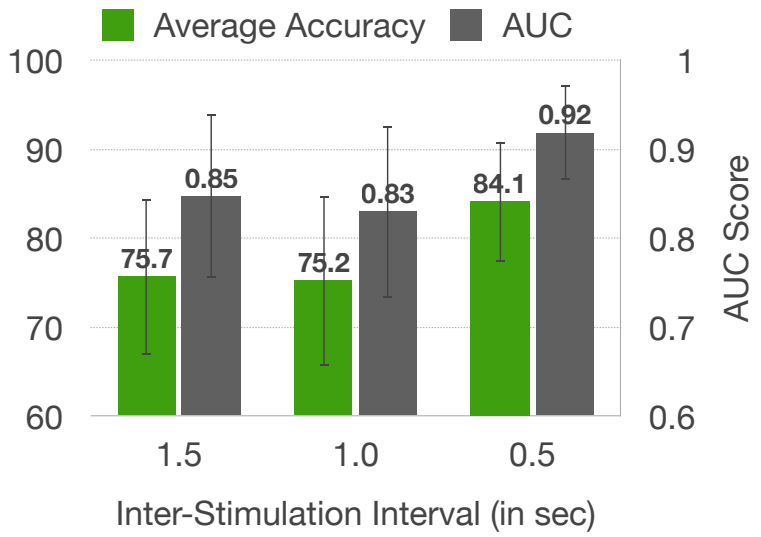


Figure 6.18: ErrP performance across Inter-Stimulation Interval (ISI)

improvement). Since the AUC score is independent of the classification threshold, we can see that the AUC score of the proposed algorithm for various values of p_{th} is similar. A similar trend is observed from Figure 6.14 in the F1 scores with over 20% increase on average on all three games. A significant improvement in the average F1 score is seen in the game Maze, where the performance increases from 51% to over 75%. It is also noted that the standard deviation of F1 scores for the proposed algorithm (14.6% overall at $p_{th} = 0.7$) is higher in comparison with the baseline algorithm (8.4% overall) hinting at a more variable performance.

6.4.4 An in-depth study of error-potentials

In this subsection, we analyze the effect of experimental variables on the quality of obtained error-potentials. Specifically, we experimentally evaluate if (a) the speed of the game or (b) the frequency of agent making incorrect actions has any direct impact on the decoding performance of error-potentials.

Effect of Inter-Stimulation Interval (ISI): Inter-stimulation interval is defined as the time duration between two consecutive actions taken by the AI agent in the given game environment. In all the experiments described above, the agent took actions every 1.5 seconds,

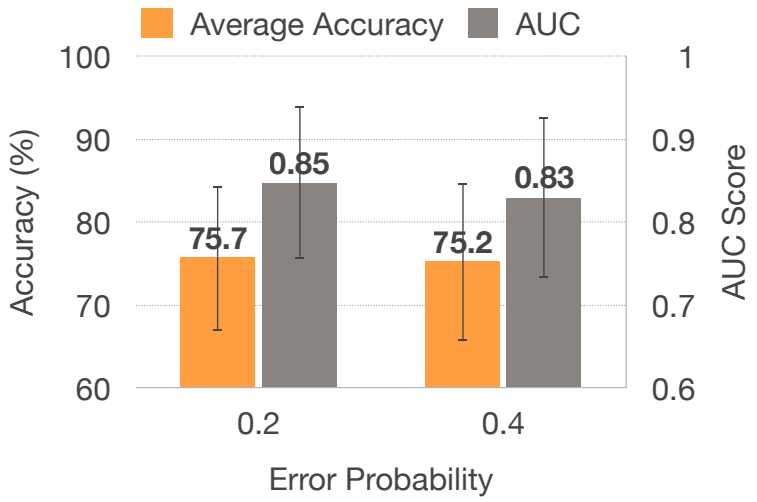


Figure 6.19: ErrP performance across agent's error-probability (P_{err})

i.e., an ISI of 1500ms. We configured the environment and repeated the experiments with the ISI of 1.0s and 0.5s. In Figure 6.18, we present the decoding performance of error-potentials for the ISI of 1.5s, 1.0s and 0.5s, in terms of accuracy and AUC score. The performance for ISI of 1.5s and 1.0s are very similar, 75.67% ($\pm$ 8.65%) and 75.17 ($\pm$ 9.51%) for accuracy, and, 0.84 ($\pm$ 0.09) 0.83 ($\pm$ 0.09) for the AUC score. Interestingly, the performance with 0.5s of ISI is significantly high, with an average accuracy of 84.1% ($\pm$ 6.78%) and the AUC score of 0.91 ($\pm$ 0.05). However, the total number of distinct experiments performed for 0.5s ISI is significantly less ($N = 2$) than 1.5s ISI ($N = 26$) and 1.0s ISI ($N = 13$), hence, further experiments are needed to support the presented claim.

Effect of trigger error probability (P_{err}): Error probability is defined as the expected percentage of incorrect moves made by the computer agent. In the previous experiments, the value of P_{err} was set to default as 0.2. We conducted additional experiments for $P_{err} = 0.4$, and compared the performance of detection of error-potentials in Figure 6.19. The detection performance for error probability was found to be similar, i.e. 75.67% ($\pm$ 8.65%) and 75.17% ($\pm$ 9.51%) for error probability of 0.2 and 0.4, respectively. Similarly, the respective AUC score is found to be 0.85 ($\pm$ 0.09) and 0.83 ($\pm$ 0.09).

6.5 Integrating RL algorithms with ErrP based Feedback

In this section, we discuss the methodologies to integrate the human feedback (obtained via error-potentials) with the reinforcement learning algorithms. Knox et al. [272] proposed basic frameworks to integrate the human feedback in any RL algorithm driven by action-value function (or Q-value function) learning. The Q-learning updates are given as,

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{6.13}$$

The agent's policy based on the Q-values is greedy as given in eq. 6.8. We define the human feebdack (obtained via eeg-based error-potentials) as follows,

$$\hat{H}(s_t, a_t) = \begin{cases} -1 & \text{if ErrP present} \\ 0 & \text{otherwise} \end{cases} \tag{6.14}$$

According to [272], the human feedback ($\hat{H}$) can be intergrated with RL during action-selection step in Q-learning in two different ways,

- **Action Biasing:** During action selection from estimated Q-value, we select actions from the modified Q-values ($Q'(s_t, a_t)$),

  $$Q'(s_t, a_t) \leftarrow Q(s_t, a_t) + \beta \times \hat{H}(s_t, a_t) \tag{6.15}$$

  The modification is performed only during the action selection, and not while updating the Q-learning values.

- **Control Sharing:** In control sharing, the probability of selecting actions is influenced as follows,

  $$P(a_t = argmax[\hat{H}(s_t, a_t)]) = min(\beta, 1), \text{otherwise base RL agent action selection} \tag{6.16}$$

Here, β is a parameter that is exponentially decayed over time.

6.5.1 Reward shaping[1]

A very naive and heavily used approach to integrate any external feedback with RL algorithms is reward shaping. In reward shaping, additional rewards are provided to augment the environmental rewards enabling the RL agent to learn optimal behavior in an accelerated manner. This allows the RL agent to deduce the optimality of the actions taken, especially during the early training process. In this context, if the goal of the RL algorithm is to learn the environment with MDP, $M = (S, A, T, \gamma, R)$, and R' refers to the additional reward function (obtained via human feedback or any other external means), the RL algorithm is trained on MDP, $M = (S, A, T, \gamma, R + R')$. In other words, at time t, the training agent receives the reward $r_t + r'_t$ instead of r_t. As we saw in the previous subsection, learning in DQN is based on off-policy Temporal Difference (TD) control, where the loss function is given by,

$$L_i(\theta_i) = E[\mathbf{r_t} + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1}, \theta_i^-) - Q(s_t, a_t, \theta_i)] \tag{6.17}$$

In DQN or any other Q-learning based RL training, reward shaping is achieved by transforming the loss function as,

$$L'_i(\theta_i) = E[\mathbf{r_t} + \mathbf{r'_t} + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1}, \theta_i^-) - Q(s_t, a_t, \theta_i)] \tag{6.18}$$

Under certain circumstances, reward shaping preserves the optimality of the learned policy [320].

In the context of Maze game, we define r'_t as below,

$$r'_t = \begin{cases} -\delta & \text{if ErrP detected} \\ 0 & \text{otherwise} \end{cases} \tag{6.19}$$

We set δ as 0.75 for evaluation purposes. Since this approach requires human feedback on every state-action pair while training, we call it as *full-access* approach. *Full-access* approach has the highest convergence rate, however, requests a large number of queries to be labeled from the human observer (via their error-potentials).

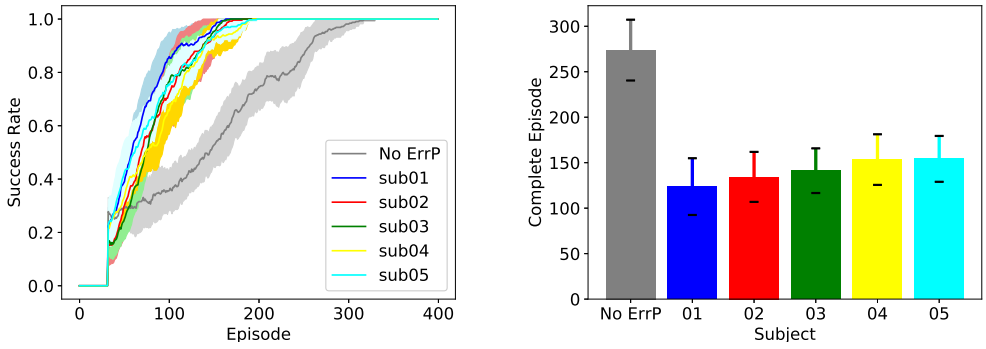


Figure 6.20: RL with *full access* to ErrP feedback

6.5.2 Evaluation[1]

In this subsection, we evaluate the training performance of the RL agent when integrated with implicit human feedback (obtained via error-potentials). We evaluate the performance for reward shaping or *full-access* method, and compare them with the action biasing and control sharing approaches. We evaluate and present the training acceleration of the RL agent on the Maze game for 5 users in terms of *success rate* and *complete episodes*. The success rate is used to measure the convergence rate of the RL algorithms. The success rate is defined as the ratio of successful plays (win) in the last 32 episodes. Complete episode is another metric we use to measure the convergence rate of the RL algorithms. The training converges and terminates at *complete episode*, when the success rate reaches to 1. We used Bayesian Deep-Q Network, BDQN as the reinforcement learning algorithm to train the RL agent [321].

We present the evaluation performance for the reward shaping based *full-access* method in Figure 6.20. "No ErrP" refers to the training of the RL agent without any human feedback. From Figure 6.20 (left), we can see that the training performance with ErrP for all 5 subjects is significantly accelerated as compared to the "No ErrP" case. Without human feedback (i.e., "No ErrP"), the agent takes 274.63 ($\pm$ 34.11) episodes to learn the optimal policy. With human feedback from S01, the number of episodes reduces to 124.67 ($\pm$ 31.49) episodes, achieving an acceleration of 2.20x. Averaged over 5 subjects, the number

Table 6.7: Number of queries for reward shaping

Subject	01	02	03	04	05
# Queries	1879.4	2072.1	2293.7	1975.4	2130.1

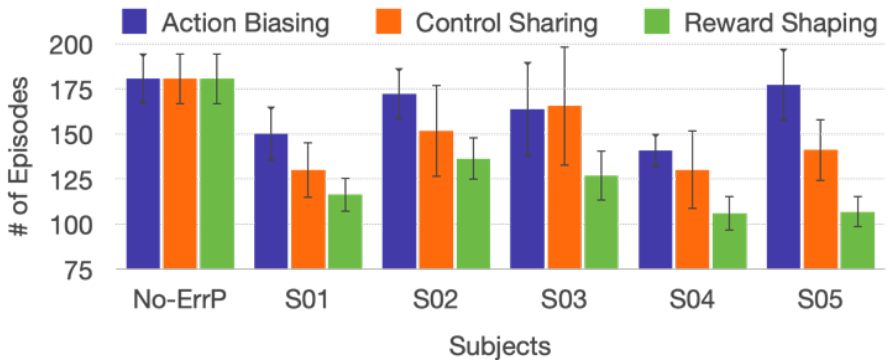


Figure 6.21: Comparing the acceleration performance on Q-learning with epsilon-greedy

of episodes required are 142.04 ($\pm$ 12.51), amounting to the acceleration of 1.94x. Since the implicit human feedback is provided on every state-action pair, a very high number of queries are made to get labeled from the human observer via their error-potentials. The total number of queries requested for the human feedback for each subject is given in Table 6.8. Averaged over 5 subjects, 2070.14 ($\pm$ 140.67) queries were requested from the implicit feedback from human subjects.

Evaluation on Q-learning with epsilon-greedy: We also evaluate and compare the human augmentation performance of action biasing and control sharing with reward shaping, and present in Figure 6.21. Averaged over 5 subjects, action biasing and control sharing achieve 1.125x and 1.25x acceleration respectively, while reward shaping performs with an average acceleration of 1.52x.

Further, for the learning curve shown in Figure 6.22, we fit $1-e^{-\lambda t}$ curve, and compare the λ and slope (i.e., $\lambda e^{-\lambda t}$) for learning with and without human feedback in the loop. For non-errp case, we obtained λ as 0.011, and slope at 40 episodes as 0.007. For S04 (highest convergence rate), the obtained λ and slope is 0.0221 and 0.0091, exhibiting a clear increase in the learning rate.

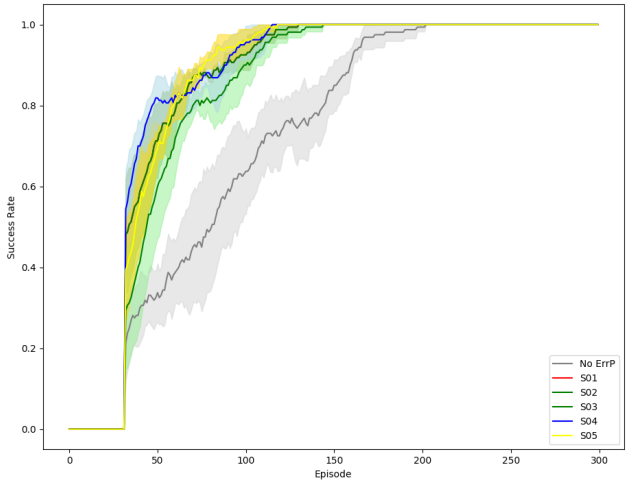


Figure 6.22: Learning curve for reward shaping with epsilon-greedy

6.5.3 Sensitivity analysis

We evaluate the importance of reliable detection of error-potential with the goal of accelerating the convergence rate of the RL algorithm for the Maze game. As described before, a negative penalty is provided as the auxiliary feedback to the RL agent upon the detection of an error-potential. The reliability of the detection of such auxiliary feedback (i.e., error-potential) is detrimental to the convergence rate of the RL algorithm. Incorrect detection of error-potential leads to noisy feedback to the RL algorithm, which could confuse the agent in determining the optimality of actions if the magnitude of the noisy feedback is really high. We run a simulation-based sensitivity analysis to quantitatively evaluate the convergence rate in the presence of noisy feedback. Specifically, we design an artificial (and external) oracle to simulate the auxiliary feedback (in the form of error-potentials) with a given accuracy rate. We train the RL algorithm to measure the number of complete episodes taken by the RL algorithm to converge to the optimal policy. We present the sensitivity analysis in Figure 6.23 for reward shaping based *full-access* method.

Without the presence of human feedback, the RL algorithm takes 170.8 episodes to converge. It can be seen from the figure, that an accuracy rate below 65% results in in-

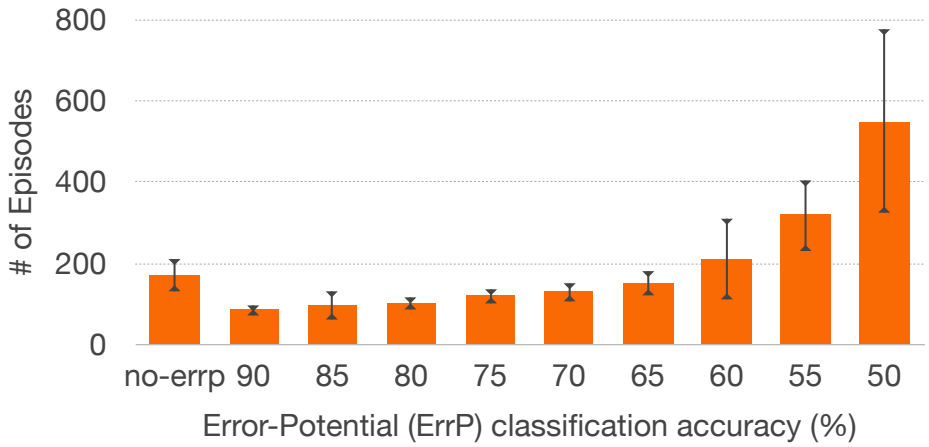


Figure 6.23: Sensitivity analysis for full-access method on Maze game: using Bayesian DQN

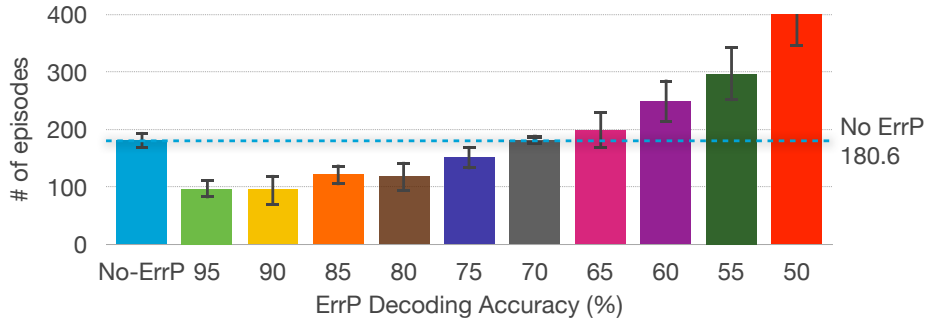


Figure 6.24: Sensitivity analysis for full-access method on Maze game: using DQN w/ epislon-greedy

creasing the number of complete episodes, and hence, reducing the training convergence rate. At 68% accuracy of error-potential detection (performance of the baseline algorithm), the number of episodes decreases to 129.4 achieving a training acceleration of 1.22x. For 80% decoding accuracy (similar to the proposed algorithm), the number of episodes reduces further to 102.2 episodes, with an acceleration of approximately 1.8x. Similarly, the sensitivity analysis on Q-learning with epsilon-greedy is presented in Figure 6.24.

6.6 Transfer Learning of Error-Potentials

The algorithms for detection of error-potentials (discussed in section 6.4) are trained in a supervised manner. Specifically, labeled examples for each state-action pair are obtained

in the format of the presence of error-potentials (if the action is sub-optimal) or absence of error-potentials (if the action is optimal). However, such information (i.e., whether an action is optimal or sub-optimal in the given state) is not available for the novel or unseen environments. This poses severe practicality issues with the applicability of the proposed framework for novel environments, since labeled examples for error-potentials are not available to train the detection algorithms. Hence, for both practicality and efficiency purposes, it is desirable to explore if ErrP detection can be learned in one setting and the learning is transferred across game environments.

We adopt a solution approach of transfer learning, where we obtain the samples from known (or seen) environments to train the classification algorithms, and use the trained classier as-is on the new (or unseen) environments without requiring re-learning of the ErrP. Particularly in our work, we assume that we know the optimal actions for the Wobble and Catch game, and use the labels to train our ErrP classification algorithm. Now for a novel or complex environment (e.g., Maze), we use the already trained classifier to infer the presence or absence of error-potentials. This is notably different from previous approaches [180, 48], where the labeled ErrPs are obtained in the same environment (where the RL task is performed). For any new and unseen environment, it does not require the human to go through the training phase again and assumes no prior knowledge about the optimal state-action pairs of the environment.

We make the case for the generalizability of the ErrP waveforms owing to their universality across humans and other primates in section 6.3.1. We observe that the manifestation of these potentials across these paradigms are found quite similar in terms of their general shape, negative and positive peak latency, and frequency characteristics [161, 180]. This prompts us to explore the consistency of the error-potentials across different environments (i.e., games, in our case) within the *observation ErrPs*. In Figure 6.10, we plot the grand average waveforms across three environments (Maze, Catch, and Wobble), to visually validate the consistency of potentials. We can see that the shape of negativity and the peak

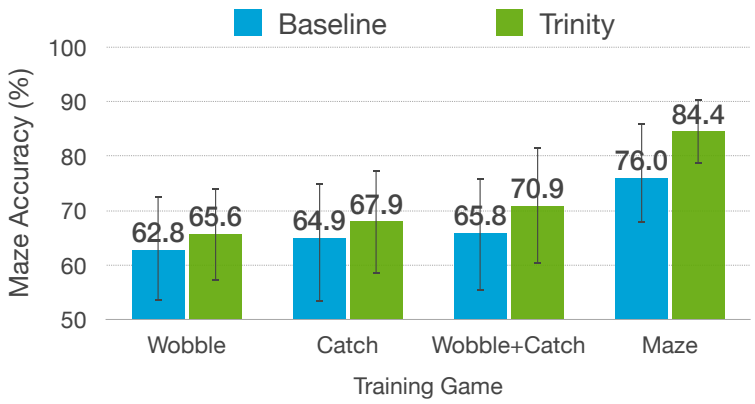


Figure 6.25: Generalizability of error-potentials: comparison of baseline and *Trinity* algorithm when tested on Maze game

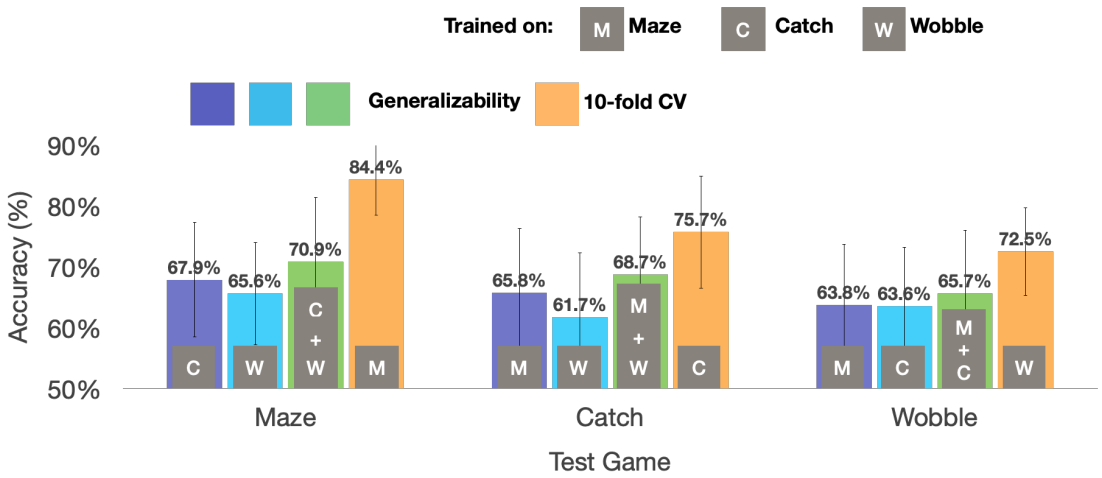


Figure 6.26: Generalizability of error-potentials: Combinations of all 3-games compared with 10-fold cross validation performance

latency is quite consistent across the three game environments. We show that by training error-potentials on one game, we can cover the variability of error potentials in other games as well which suggests that error-potentials are indeed generalizable across environments, and can further be used to inform deep reinforcement learning algorithm in new and unseen environments.

6.6.1 Evaluation

To evaluate the transfer learning (i.e., generalization capability) of error-potentials and the decoding algorithm, we train on the samples collected from the Catch game and test on the

Maze game. As Catch is a simple game, we assume the optimal action for each state is already known (providing the labeled examples to train the ErrP decoder). Since information about state-action optimality is given for the Catch game (can be assumed for simpler game environments), and thus labeled examples are obtained for the Catch game to train the ErrP decoder. However, the Maze game needed to be solved, hence, we do not make any assumptions about the optimality of the actions. In Figure 6.25, we compare the detection accuracy of the baseline and proposed *Trinity* algorithm over the Maze game for 6 subjects. For 10-fold cross-validation (i.e., train and test on Maze), the proposed algorithm performs with an accuracy of 84.4% ($\pm$ 5.91 %). The 10-fold cross-validation scheme serves as an upper bound for the generalizability performance. When trained on samples from both Wobble and Catch games, the proposed algorithm performs with an accuracy of 70.86% ($\pm$ 10.65%), an improvement of 7.75% over the baseline algorithm. When trained individually on Wobble and Catch game, the proposed algorithm performs with an average accuracy of 65.63% ($\pm$ 8.41 %) and 67.9% ($\pm$ 9.47 %) respectively. When trained on samples using both Wobble and Catch game, the algorithm is able to capture the 84% variability in the decoding of error-potentials for the Maze game.

We also present the generalizability performance over all combinations of the game environments for the proposed algorithm in Figure 6.26. It should be noted that a 10-fold cross-validation scheme is used for the evaluation part with the same game environment employed for training and testing (i.e., orange bars in Figure 6.26), and serve as an upper bound for the generalization performance. We can see from the figure that generalizability performance increases when trained on samples from two games instead of on a single game. For Maze, the decoding accuracy is 70.86% ($\pm$ 10.65%) when trained on both games, namely Wobble and Catch. The Maze performance drops to 67.90% ($\pm$ 9.47%) and 65.63% ($\pm$ 8.41%) when trained individually on the Catch and Wobble game. Similarly, when tested on Catch, the decoding accuracy is 68.7% ($\pm$ 9.55%) when trained on both Maze and Wobble game. Compared across three games, Maze has the highest accuracy

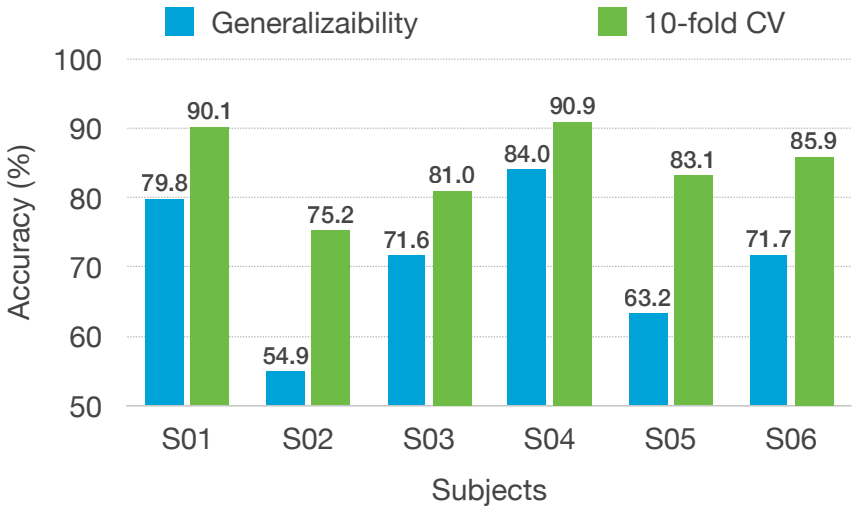


Figure 6.27: ErrP decoding accuracy: across subjects

pertaining to its intuitive and user-friendly design and short duration experimental trials.

These experiments validate that the error-potentials can be learned in a generalizable manner to avoid re-training of the human feedback (via EEG) decoder.

Performance across users

In this subsection, we analyze error-potential performance across human subjects. We present the accuracy of ErrP detection individually for 6 subjects in Figure 6.27. In this analysis, the subjects are evaluated on samples from Maze games, when trained on samples from both games Wobble/Catch (generalizability) and on Maze itself (10-fold cross-validation scheme). Subject 01 and 04 have the highest 10-fold CV accuracy of above 90.1% and 90.9%, while Subject 02 has the lowest 10-fold CV accuracy of 75.2%. We obtain a 10.6% standard deviation across six subjects. For generalizability performance (i.e. when the algorithm is trained on samples from both Catch and Wobble game and tested on samples from Maze game), subject 04 presents the highest accuracy of 84.0%, while subject 02 has the detection performance close to random, 54.9%. We obtained a standard deviation of 5.91% for the generalizability performance across six subjects. The high variability in the transfer learning performance could be due to factors including significant change of electrode cap placements across sessions, affective and environmental state of

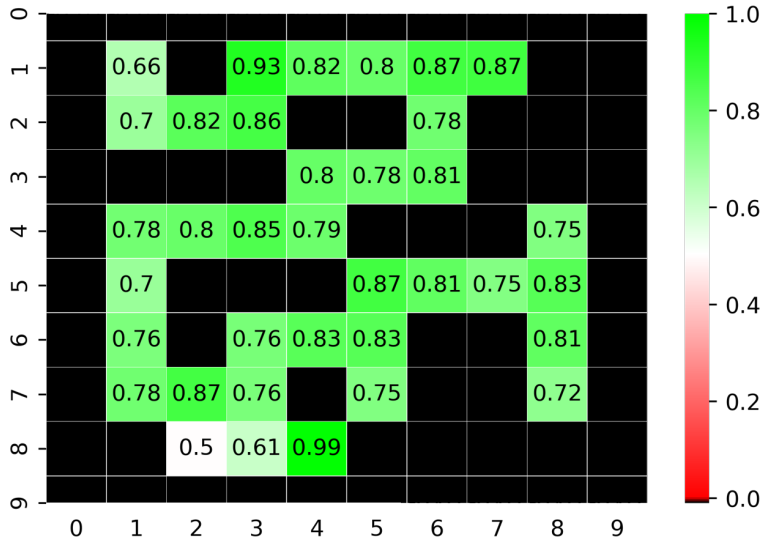


Figure 6.28: ErrP decoding accuracy: across Maze states

the subject, etc.

Maze performance across states

We also analyze the average performance of 6 subjects across different states of the Maze game, and present in Figure 6.28. The performance across the states is quite consistent. It should be noted that in one particular state (row: 8, col: 2), we have two optimal actions (moving UP, and RIGHT), and hence the ErrP detection accuracy is very close to random. On averaging accuracy numbers over all possible states in the Maze game, we receive an aggregate accuracy of 78.91% ($\pm$ 8.65).

Performance over importance of errors

In the maze game, there are two types of errors possible,

- Errors of commission: The agent makes an incorrect move to a new cell
- Errors of omission: The agent makes an incorrect move where it hits a wall, such that the final position of the agent remains same.

We analyze the difference in detection accuracy of error-potentials individually for errors of commission and omission. For such analysis, we train our classification algorithm on

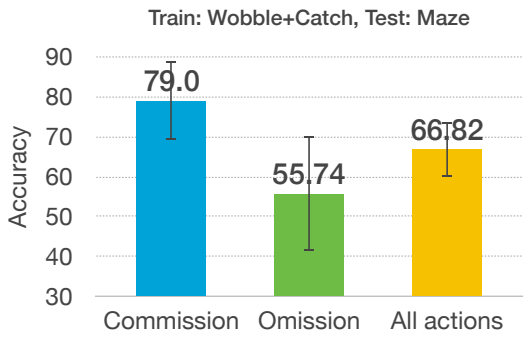


Figure 6.29: ErrP accuracy for commission and omission errors

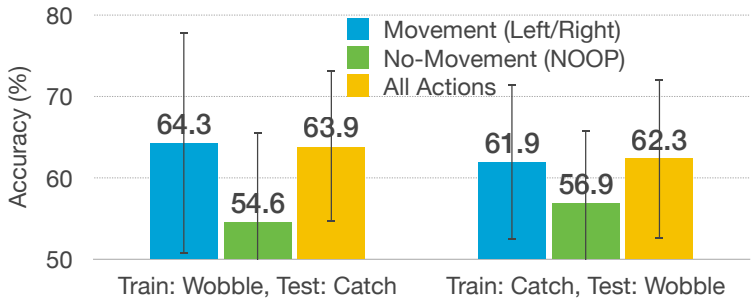


Figure 6.30: Transfer learning from movement (left/right) to no-movement (NOOP)

the samples obtained from the Wobble and Catch game, and use samples of Maze game for evaluation purposes (Figure 6.29). The average accuracy for all actions is 66.82% ($\pm$ 6.89). For commission errors, the average accuracy is 79.02% ($\pm$), while the omission errors have accuracy of 55.74% ($\pm$ 14.20).

In the context of the Maze game, the commission errors adds two step lags (i.e., the agent has to take two additional steps to finish the game), while the omission errors add only one step lag (since the agent is in the same position). Hence, the commission errors are more important than omission errors, which reflects accordingly in the ErrP detection results.

Transfer learning over actions

We study the transfer learning (or generalizability) in the action space. Specifically, we train our classification algorithms on the samples obtained from the Wobble game, and compute the difference in accuracy on catch game for two cases - (a) when movement

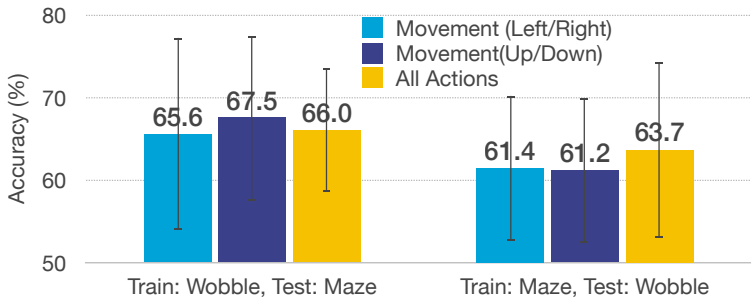


Figure 6.31: Transfer learning from horizontal (left/Right) to vertical (up/down) movements

actions are considered (Left/Right), and (b) when only no-movement (i.e., no-operation) action was considered. Recall from the previous section that in the Wobble game, the agent can take only left and right actions, while the agent in the Catch game can take three different actions (moving to the left, moving to the right, or stay in the same grid). From this analysis, we are particularly interested in studying if the error-potentials are transferred from one set of actions to a different set of actions across environments. We present the results of this analysis in Figure 6.30. The aggregated accuracy for the Catch game (for all actions) is 63.86% ($\pm$ 9.34). The accuracy for movement actions (i.e. left, and right) is 64.26% ($\pm$ 13.62), while for non-movement actions (i.e., no-operation) is 54.56% ($\pm$ 11.0). The significant accuracy difference in the two cases indicates that the movement to non-movement actions are not easily transferable. We notice a similar trend in Figure 6.30 when the classification algorithm is trained on the Catch game, and tested on the Wobble game.

We perform a similar analysis on the Maze game (trained on Wobble game) and divide the two cases as (a) moving left or right, and (b) moving up or down. Since the classification algorithm is trained on the samples obtained from the Wobble game (where the agent takes left and right actions), we are interested in studying if the one set of movement directions (i.e. left and right) are transferable to other set of movement directions (i.e., up and down). The accuracy over all actions was obtained as 66.03% ($\pm$ 7.44). For the horizontal actions (i.e., left and right), and vertical actions (i.e., up and down) we received the accuracy of

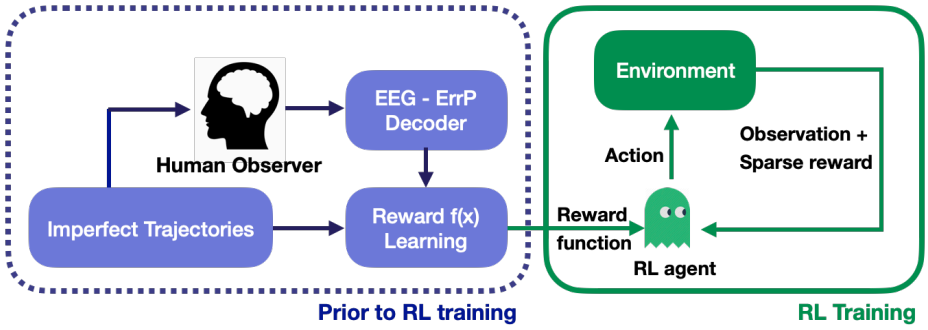


Figure 6.32: Functional architecture of combining human feedback with RL algorithm through imperfect demonstrations

65.6% ($\pm$ 11.6), and 67.5% ($\pm$ 9.9) respectively. We obtained similar results on flipping the training and testing dataset (i.e., training on Maze game, and testing on Wobble game).

From the above analysis, we conclude that movements to non-movements are not easily transferable, however, different movements are easily transferable.

6.7 Learning from Imperfect Demonstration[1] for RL integration

Our collaborators proposed a practical approach to combine human feedback with RL algorithms in an efficient manner, i.e. without querying every state-action pair for human feedback. The proposed approach is derived from the principle of *learning from imperfect demonstrations*, where a quality function (Q) is learned acting as the proxy for implicit human feedback. Specifically, prior to the training of the RL algorithm, the humans are shown the imperfect trajectories (i.e., the trajectory followed by the agent includes sub-optimal behavior) of an agent playing the game. These trajectories are designed with the help of an expert human. Human subjects are asked to silently observe and assess the actions of the agent in the shown imperfect trajectories, and their brainwaves are simultaneously recorded to estimate error-potentials. With the estimated error-potentials, a quality function (Q) is learned offline (and prior to the training of the RL algorithm). Further, based on the quality function, an alternative reward function (r') is learned, acting as a proxy for the human feedback to augment the environmental rewards, during the training of the RL algorithm as shown in Figure 6.32.

Table 6.8: Comparing number of queries for the human feedback for integration frameworks - full access method, and method based on imperfect demonstrations

Subject	01	02	03	04	05
Full access	1879.4	2072.1	2293.7	1975.4	2130.1
Imperfect demonstrations (20 trajectories)	505.7	394.7	587.1	681.4	361.3

Since the human feedback is queried on the state-action pairs encountered in the trajectory, and prior to the RL algorithm training, the number of queries are significantly less as compared to the *full-access* method.

The alternative reward function is learned using the maximum entropy policy learning principles [322, 323]. In the maximal entropy learning principle, the goal is to maximize the cumulative discounted sum of rewards and entropy, i.e.,

$$\pi_{entropy} = \text{argmax}_{\pi} \sum_{t} \gamma^{t} [R_t + \alpha H(\pi(. \mid s_t))] \tag{6.20}$$

leading to the optimal policy ($\pi*$) as follows [322],

$$\pi*(a \mid s) = exp((Q*(s,a) - V*(s))/\alpha) \tag{6.21}$$

where $Q*(s,a)$ is the state-action value function of the optimal policy, and optimal state-value function ($V*(s)$) is given by,

$$V*(s) = \alpha log \sum_{a} exp(Q*(s,a)/\alpha) \tag{6.22}$$

For the learned policy π, the Q function is trained by applying Maximum Likelihood (ML) on positive state-actions pairs ($\pi_Q(a \mid s)$) and negative state-action pairs (1 - $\pi_Q(a \mid s)$). Further, a baseline function $t(s)$ is added, to stabilize the training of the Q function.

6.7.1 Evaluation

For the proposed framework based on learning from imperfect demonstrations, the training performance is provided in Figure 6.33 and 6.34, respectively for 10 and 20 trajectories (shown to the user prior to the RL training). For 10 trajectories, it took 111.67 ($\pm$ 37.86)

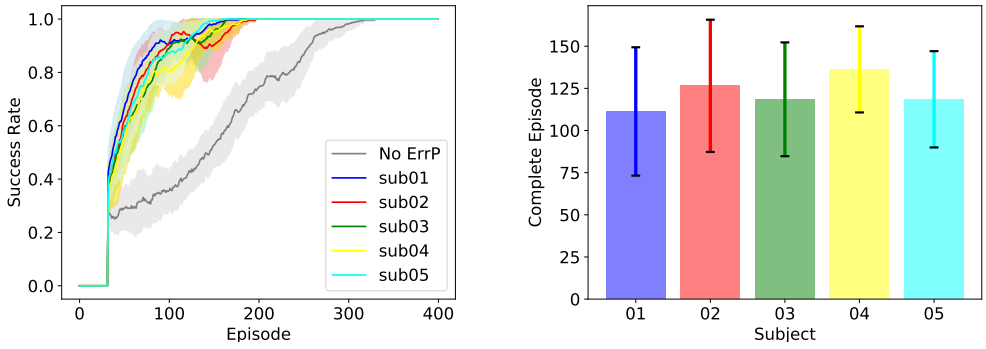


Figure 6.33: RL with proposed framework: learning with imperfect demonstrations on 10 trajectories

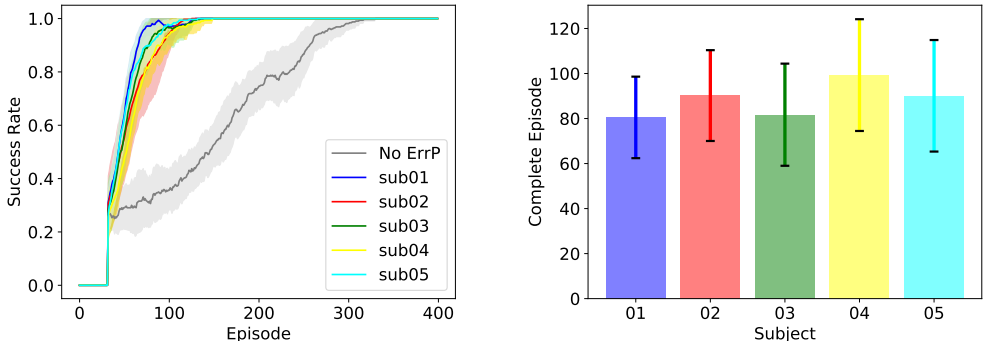


Figure 6.34: RL with proposed framework: learning with imperfect demonstrations on 20 trajectories

episodes to learn the optimal policy based on feedback from S01, achieving an acceleration of 2.46x. Averaged over 5 subjects, the optimal policy was learned in 122.55 ($\pm$ 9.35) episodes, with an average acceleration of 2.25x ($\pm$ 0.167x). The acceleration further boosted with 20 trajectories, converging in 81.17 ($\pm$ 17.65) episodes, an acceleration of 3.38x for S01 feedback. Averaged over the 5 subjects, the RL algorithm converged in 88.93 ($\pm$ 7.50) episodes, with an average acceleration of 3.28x ($\pm$ 0.43x). We also present the number of queries required for the 20 trajectories in Table 6.8, and compare them with the full-access method. Averaged over the 5 subjects, the number of queries made were 506.04 ($\pm$ 118.88), approximately 75.56% less as compared to the *full-access* method.

6.8 Summary

We researched and developed a framework to tackle the slow convergence issue with RL algorithms by introducing implicit human feedback in the loop obtained via EEG-based error-potentials. We conducted IRB approved user trials where users observe an agent learning to play the games and their ErrP signals are being monitored using an EEG headset. We then use the dataset to show that there is a strong inverse correlation between the human observer's ErrP signal and the correctness of the agent's actions, thus validating the candidacy of ErrP as a potential feedback signal for the reinforcement learning algorithms. We present *Trinity*, an error-potential decoding algorithm leveraging multi-dimensional aspects of the EEG (namely, spatial, frequency, and time-domain) to increase the accuracy of detecting ErrP. We then integrate the ErrP signals into the reward function of the RL algorithm and study the acceleration achieved with respect to the algorithm's convergence time to a success rate of 1. We show that significant acceleration can be achieved by the integration of human feedback with the default reward function that the game provides. We study the transfer learning of error-potentials over environments, and actions, removing the requirement of obtaining labeled training examples, and hence the system can be used for unseen and novel environments. Further, an advanced approach is discussed to improve the acceleration in convergence rate while reducing the number of queries made for the implicit feedback.

6.9 Appendix I: Experimental Protocol

In this section, we explain our experimental protocol collection of EEG data (specifically, error-related potentials) of human subjectsin detail.

Material list

- EEG Acquisition Software (OpenBCI–GUI[10] and OpenViBE[11]). Both softwares are available to download for free.
- Hardware (OpenBCI Cyton Board)
- Electrode Cap (BIOPAC-CAP 100C), and 2 Ag-AgCl ear-clip electrodes
- 1 Computer System (We use Linux environment)
- Electrolyte Gel (BIOPAC Electro-Gel)
- Chest Strap Band and Plastic Syringe (without needle)
- Tissues
- Human Participants

Preparation [before the subject setup]

- Make sure the cap, ear electrodes and syringe is clean.
- Make sure that the transmitter (Tx) and receiver (Rx) of OpenBCI are kept at a minimum distance from each other (ideally the Tx and Rx modules should be placed next to each other)
- Setup the connections of the system.
  - Connect OpenBCI channels 1-16 to Electrode Cap channels.
    - Fp1, Fp2, Fpz, F7, F3, Fz, F4, F8, C3, Cz, C4, P3, Pz, P4, O1, and O2
  - Connect Reference (or SRB2 pin) of OpenBCI with white ear electrode
  - Connect BIAS pin of OpenBCI with blue ear electrode

[10]https://docs.openbci.com/docs/06Software/01-OpenBCISoftware/GUIDocs
[11]http://openvibe.inria.fr/

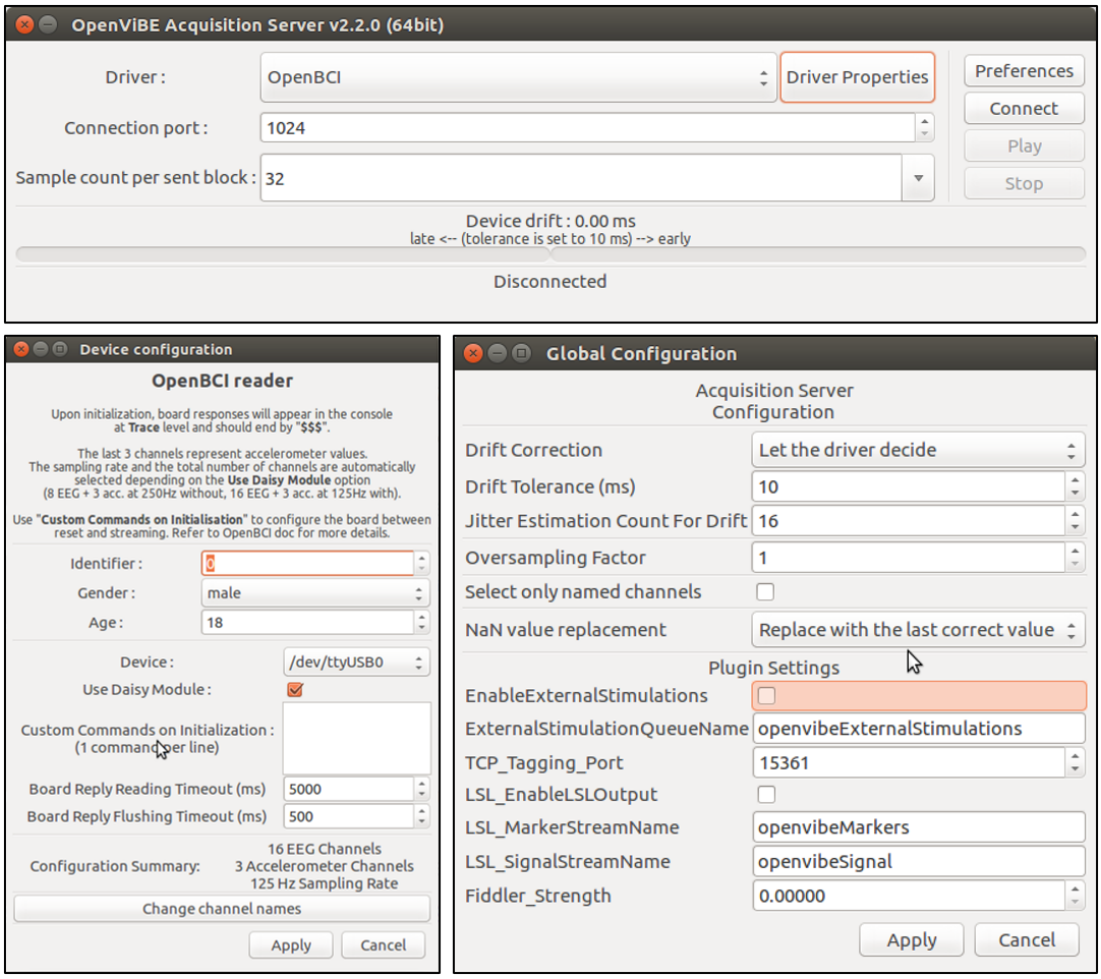


Figure 6.35: OpenViBE settings

- Design experimental paradigm such that
  - The trials are short (100s). This allows subjects to remain focused during the trial.
  - One experiment should contain multiple trials, allowing to collect a large number of data samples.
  - Add the 10 seconds delay in the script to allow them time to setup the screen.

Subject preparation

- Explain the experimental protocol to the subject, obtain and document the informed consent of the research subjects. It is imperative to use the IRB approved consent

forms for explaining and obtaining the informed consent form.

- Ask the subject to play the game in manual mode. Further, show the demo of the game to the subject as it will be seen by the subject during data collection. This enables the subjects to get acquainted with the goals of the given game environment.
- Close lab doors and ask other lab members to remain silent and stationary during the experiment. Any external noise and interferences should be minimized to the extent possible.
- Select the correct cap size for the subject based on the scalp size and fit. The electrode cap should be snug-fitting, although not extremely tight such that it is uncomfortable for the subject.
  - We have a small and medium cap. We select the cap directly based on fitting.
- Instructions for the subject
  - Restrict the head movements as much as possible.
  - Be focused during the experiment and pay attention to the stimuli (i.e. actions taken by the computer agent). Take longer breaks if feeling fatigued or drowsy.
- Mount the cap on the human participant
  - Roughly check that the Fp1 and Fp2 locations lie directly above the eyebrows.
  - Additionally, verify the placements of O1 and O2 electrodes.
  - Use the syringe to insert a little gel inside the ear electrodes, and clip the white electrode to the left earlobe and blue electrode to the right earlobe of the human subject.
  - Secure the cap with a chest strap.

- Apply abrasive electrode gel using syringe between electrode and scalp to minimize the impedance.
  - Make sure to put 'enough gel'. Too much gel reduces the spatial locality of the EEG, and too little of the gel would fail to maintain low-impedance contact between the scalp and electrodes. If the gel is provided in excess, it tends to flow out and spread to a larger area of the scalp, reducing the spatial resolution. This results in averaged recording EEG activity over the neighboring locations (neighboring locations of the particular electrode where the gel is in excess) reducing the signal-to-noise ratio, as it will include EEG corresponding to more unwanted activities.
- Make sure the OpenBCI transmitter (Tx) module is close to the receiver (Rx) module. In OpenBCI, the Rx module is always attached to the PC. The reasoning behind the distance between Tx and Rx module is explained in detail in section 6.10. A USB extending cable can be used to decrease the distance between Tx and Rx module.
- Use OpenBCI GUI to validate the connections of the EEG cap
  - All electrodes must be around 10 uV in the rest state.
  - Ask the subject to blink their eyes. Eye-blinks should be clearly visible on the amplitude vs. time-domain EEG for Fp1 and Fp2 electrodes in the OpenBCI GUI.
  - Ask the subject to close her eyes for 3-4s, and a peak should be visible around 10Hz in the frequency spectrum. This validates the connections at O1 and O2 locations.
- Start OpenViBE acquisition server and OpenViBE designer for ErrP data collection (Make sure the settings are as shown in Figure 6.35).

- Connect and play to check the device drift. A drift of 1 correction per 8 seconds is normal (i.e., one packet offset every 8 seconds). If the device drift is significantly higher, check connections, the distance between Tx and Rx, and try to reset and re-connect the device. Further, turn off any other environmental components operating at 2.4 GHz spectrum (e.g., Microwave, other Bluetooth devices, etc.)

During the experiment

- In the OpenViBE designer, edit the filename for the raw data (".csv" in our case) and start the scenario. Visually check the signal appearance with the eye-blink test in the OpenViBE signal visualization.

- Start the python script (*agents/record_EEGdata.py*), and set up the screen (close all other monitors and background applications to remove the distractions)

Between the subject trials

- Disconnect OpenBCI every time the trial is finished, and re-connect before resuming the next trial.

- Ask the subject to be in the rest state for 30 seconds.

- Re-connect OpenBCI and ensure to change raw data filename (".csv" file) and start recording data.

After the experiments

- Check the files are saved properly and move them to a secure location.

- Anonymize the recorded data and add all metadata and notes (e.g., variables in the experiment)

- Ask the subject to fill any post-experimental survey, if there is any.

- Clean the equipment.

6.10 Appendix II: System-related Issues with Low-cost EEG-based BCIs

Event-Related Potentials (ERPs) are fluctuations or responses reflected in the EEG that are both time- locked and phase- locked to the event. The amplitude of the scalp recorded ERPs is low, making the single-trial estimation a very difficult task. Since the ERP signals are time- and phase- locked to the stimulus, N number of signals can be averaged to boost the SNR (rhythmic or quasi-rhythmic activity averages out, while time- and phase- locked activity receives achieves additive effect). Specifically, if the signal power is P, and zero-mean noise variance is σ^2, the SNR of a single-trial ERP is,

$$SNR_{old} = \frac{P}{\sigma^2} \qquad SNR_{new} = \frac{P.N}{\sigma^2} \tag{6.23}$$

Where P is the signal power, and σ^2 is the variance in noise. Such an approach is used in the ERP detection algorithms to improve the classification accuracy [324]. For example, obtaining the prototype response in the baseline ErrP detection algorithm is based on the same principle (as explained in Algorithm 3).

Since the signal detection pipeline for event-related potentials relies on the prototyping (leveraging the precise time-locking), it could be highly sensitive to the jitter time (or acquisition drift), i.e., the delay between the actual presentation of the stimulus to the user, and the time instant where the stimulus is marked with the EEG signal. If the jitter is constant, it would pose no harm to the signal processing pipeline, however, if the jitter is variable it could have significant impacts on the detection accuracy.

Low-cost EEG-based BCIs, specifically, OpenBCI uses Bluetooth Low Energy (BLE) to transmit the raw EEG signals from the OpenBCI board to the computer (or mobile) device. In our experiments, we used 125Hz sampling rate over 16 channels. In practice, 2000 samples per second over BLE causes packet loss in case of increased distance b/w Tx/Rx or due to channel interference issues. We performed an empirical study to conclude that the packet drop rate indeed increases with the distance between Tx and Rx modules. The packet drop rate is proportional to the distance. Further, [325] also concluded the

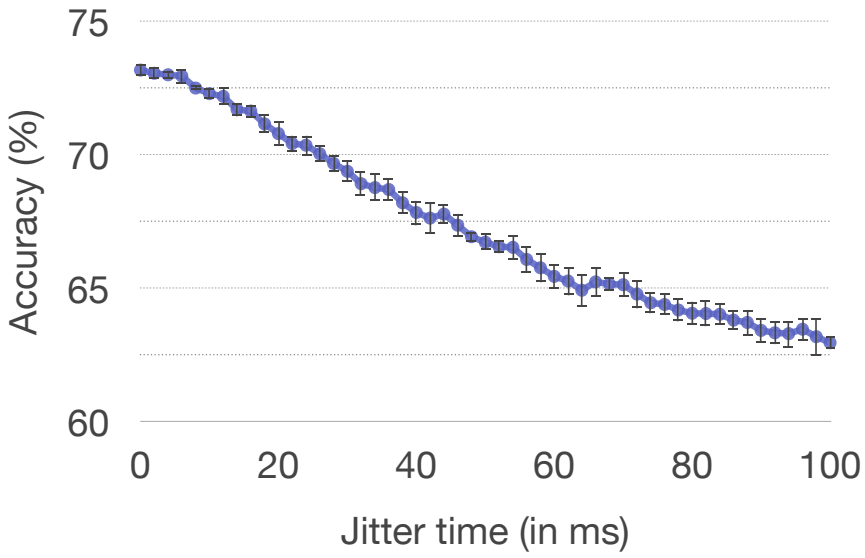


Figure 6.36: Impact of jitter on ErrP detection accuracy

delayed arrival of packets, and the presence of jitter while recording data from OpenBCI.

OpenViBE acquisition server continuously monitors the incoming sample rate (between comparing the number of samples arrived and the theoretical sample count based on the device frequency). If the difference between the number of arrived packets and the ideal count is significant, the OpenViBE acquisition server initiates a drift correction mechanism by adding dummy packets. This equates to the tempering of data and has serious implications on the synchronization offset between ERP signals and the stimulus markers. To quantitatively evaluate the impact of such added jitter, we performed a simulation based analysis, where for some fixed jitter interval length j_t, we generate a random offset uniformly between $[-j_t/2, +j_t/2]$, and compare the accuracy by varying the j_t. We performed monte-carlo simulations for such analysis, and present result for an average on 10 runs in Figure 6.36.

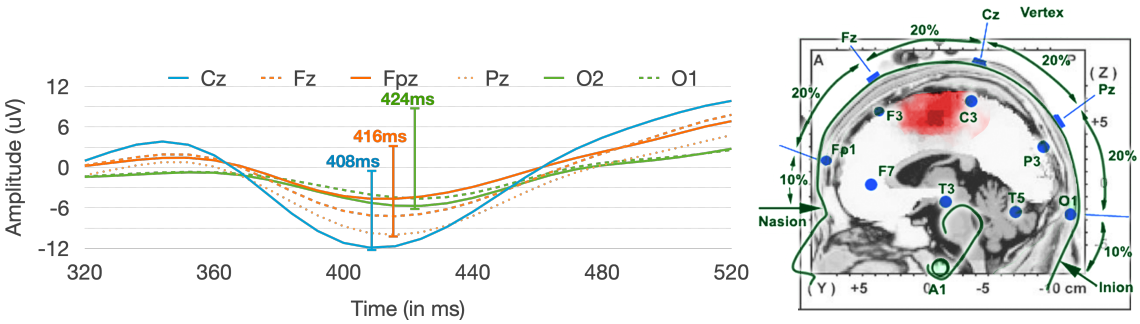
From the figure, we can see that a jitter of upto 20ms reduces the accuracy from 73.17% to 70.8%, a 3.23% reduction in accuracy for 20ms jitter. Further, an 80ms of jitter reduces the accuracy from 73.17% to 64.07%, a 12% reduction. For BLE based communication, a 20ms and 80 ms offset could be incurred by a mere dropping of 2.5 and 10 packets in one

second respectively. Since the packet drop rate was found to be proportional to the distance between the Tx and Rx module of the OpenBCI board, they should be kept at a minimum distance from each other while recording any ERP data.

6.11 Appendix III: Experimental Evidence for Error-Potentials

In this section, we first provide the visual evidence of the presence of error-potentials in our experiments. In several works, the origin of the error-potentials is believed to be from Anterior Cingulate Cortex (ACC), with the highest activity in fronto-central region (e.g., Cz, FCz, etc.) [180, 176, 178, 163, 326, 327]. Our hypothesis is that if the observed signal (error-related potentials) is originated from the Anterior Cingulate Cortex (ACC), source of error-related negativity (ERN) and error-positivity (Pe), it should arrive at the closest electrode first (Cz). The order of arrival at different electrodes should follow the same order as per the distance from the source of the signals. In [328], sLORTEA method was used to localize the source of the signals as shown in Figure 6.37b. The identified source location is marked in red color, and the EEG electrode pattern is overlayed on the brain scan image.

We analyze the data recording of Maze for subject S07 (highest accuracy set) to provide visual evidence of the presence of error-potentials in our experiments. From the particular experiment, we filter all the brain potentials corresponding to the incorrect stimulus (i.e., when the agent took an incorrect action). We analyze the Error-Related Negativity (ERN) peaks at different electrode locations in Figure 6.37a. Here, 0 seconds represents the stimulus marker. The ERN peak for Cz is recorded at a delay of 408ms. Electrodes located in the fronto (Fpz, Fz)- and parietal (Pz)- region are distant from ACC (as compared to central Cz). The ERN peak for fronto- and parietal- band is achieved at approximately 416 ms with less intensity. Additionally, inion electrode positions (O1 and O2) are more distant from the ACC as compared to the fronto-, central- or parietal- region. The ERN minima recorded at ERN is further delayed (424ms, with very less amplitude strength) validates



(a) Arrival order of error-potentials on different EEG electrodes

(b) ErrP source localization Image taken from [328]

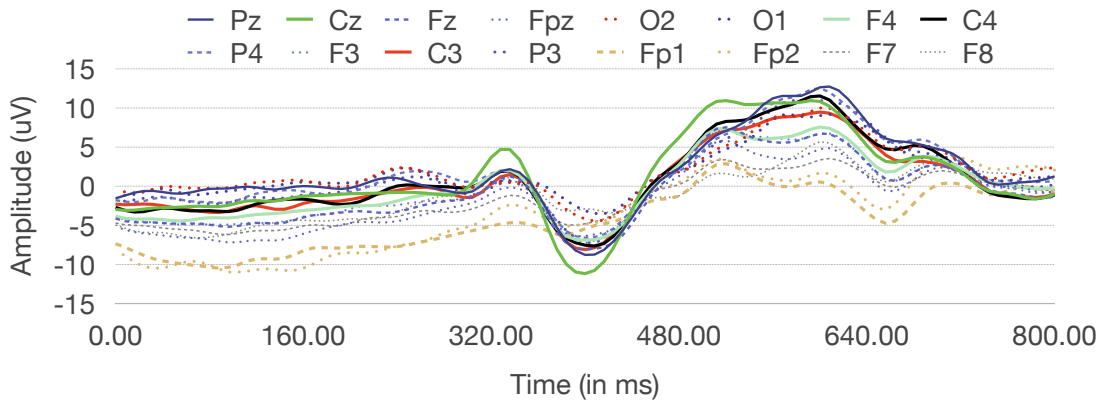
Figure 6.37: Evidence of error-potentials: computing the time instances of Error-Related Negativity (ERN) peaks at different electrode locations

our hypothesis. In summary, the above analysis provides preliminary evidence that the signals are indeed generated from the ACC, by measuring the time instant and amplitude intensity of ERN peaks recorded at different electrodes. Since the sampling frequency in our experiments is 125 Hz, we achieve a granularity of 8ms (1/125 Hz) in the time-domain.

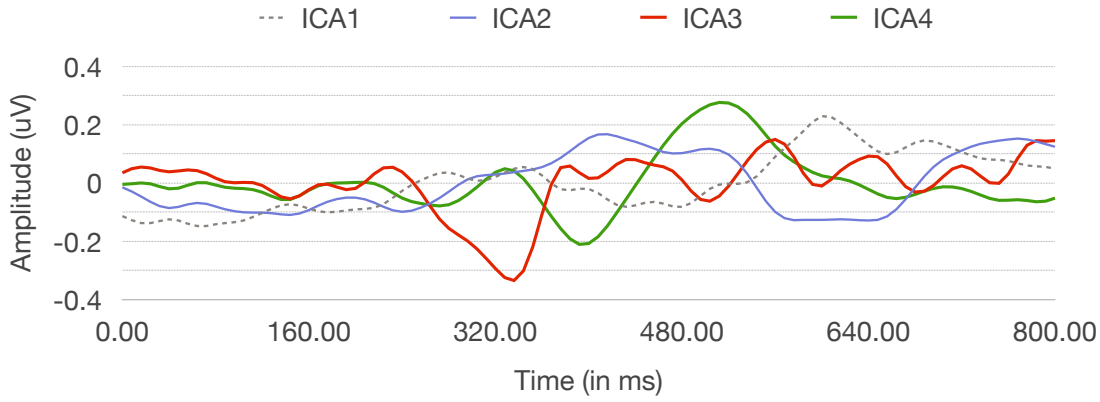
We plot the average waveforms for 16 channels for erroneous stimulus (i.e., presence of error-potentials) and non-erroneous stimulus (i.e., absence of error-potentials) in Figure 6.38a and Figure 6.39a. We apply Independent Component Analysis (ICA) to separate the averaged waveforms into the individual components. For this analysis, we decompose the signals into four additive subcomponents, and backtrack the individual components into their spatial distribution using topography maps. For each ICA component, the contribution strength from each electrode is computed to create the topographic map[12].

For error-potentials, the ICs and topographic maps are shown in Figure 6.38b and Figure 6.38c. The ICA4 component (shown in green line in Figure 6.38b) corresponds to the error-related potential with peak ERN negativity at 400ms and maximal distribution in centro-region (Cz and near electrodes). ICA3 (solid red) corresponds to the component related to the visual stimulus change (since the screen content is also changing at the stimulation time), with a peak around 320ms ($\approx$ 80ms before the ERN peak), and maximal

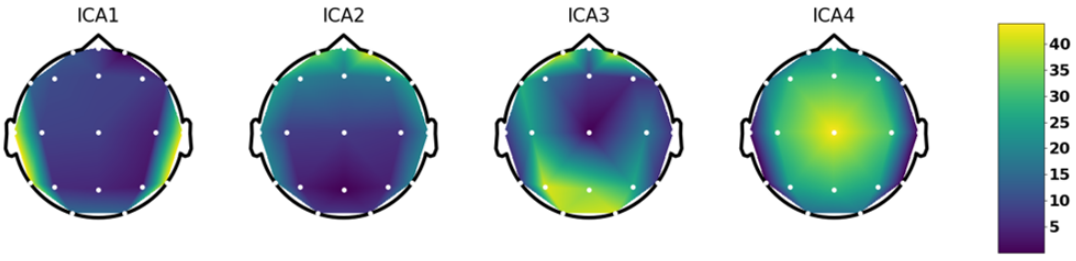
[12]We use MultiVariate Pattern Analysis (MVPA) library in Python (`http://www.pymvpa.org/`) to generate the topographic maps



(a) Average ErrP signals

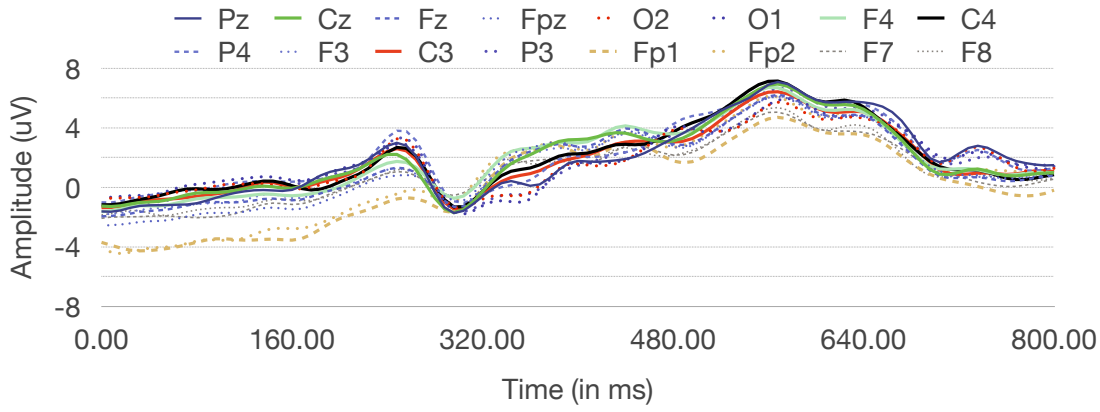


(b) Independent Components (IC) for ErrP signals

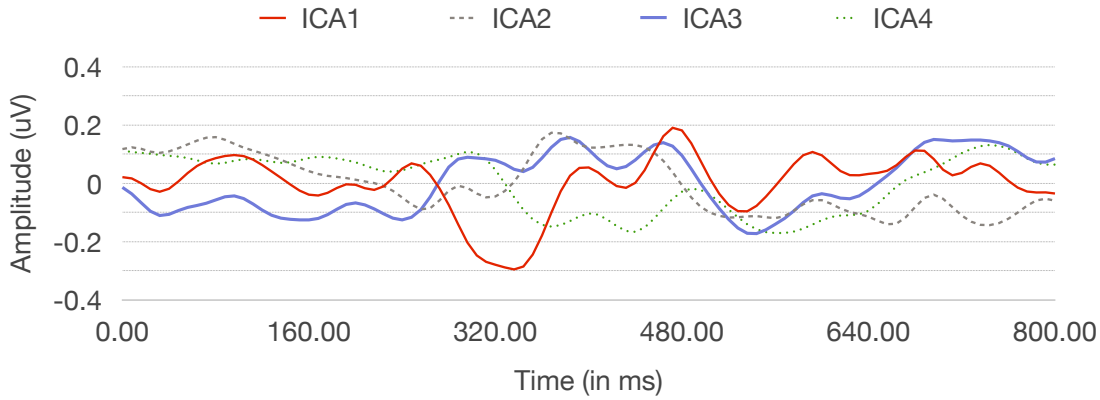


(c) Topographic Map of ICA components for ErrP signals

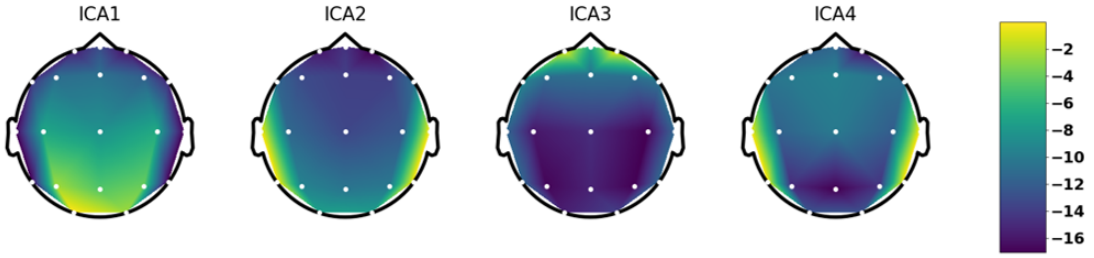
Figure 6.38: Analysis of ErrP (when agent took incorrect action) signals [subject 07: Maze game]



(a) Average Non-ErrP signals



(b) Independent Components (IC) for non-ErrP signals



(c) Topographic Map of ICA components for non-ErrP signals

Figure 6.39: Analysis of non-ErrP (when agent took correct action) signals [subject 07: Maze game]

distribution at inion electrodes (O1 and O2). The earlier peak for visual stimulus change is intuitive since the stimulus change should be processed before the error- processing. ICA2 has maximal activity at Fp1 and Fp2 electrodes, and corresponds to the eye-blinks.

We show the average waveforms for correct-stimulus (i.e., absence of error-potentials) in Figure 6.39a. For non-error potentials, the ICs and topographic maps are shown in Figure 6.39b and Figure 6.39c. A visual related IC (solid red) can be seen with maximal negativity peak at $\approx$ 320ms. The visual stimulus related IC is consistent across both error-related waveforms and non-error waveforms. An eye-blink component can be seen (ICA3, solid blue line) with the maximal activity distribution at Fp1 and Fp2 electrodes.

CHAPTER 7

CHALLENGES AND NEXT STEPS

In this thesis, we have investigated the interplay between ML algorithms and EEG-based BCIs through the lens of end-user usability. Specifically, we studied the interplay on two fronts, (a) using ML techniques to solve challenges in EEG processing, and (b) enabling human-assisted ML with EEG-based human input. First, we studied how ML can be used as a powerful tool to learn and characterize the brain activity of an individual to build meaningful applications with it for day-to-day use cases. We identify the short battery life problem with BCI wearables and perform an experimental analysis to find control knobs to switch the BCI wearable in the low-power mode. We studied the modality choice for the wake-up command design, and after careful consideration of practical benefits, select eye-blinks as the wake-up command modality. We proposed *Trance*, a wake-up command detection system to improve the battery life by 2.7 times, making the BCI wearables last for day usage. We propose *BLINK* algorithm to detect eye-blinks in a completely automatic and unsupervised manner with an accuracy rate of above 98%. Second, we propose systems and algorithms to fundamentally improve the ML algorithms using EEG-based human input or implicit feedback. We propose *Cerebro*, capable of ranking consumer products according to the user preferences by relying solely on the user's brainwaves. Then, we propose a novel paradigm to allow humans to assist learning algorithms in an implicit manner, accelerating the convergence rate of RL algorithms. For this research, we develop system and experimental protocols to conduct human studies and propose an algorithm to detect implicit human feedback reliably in the form of error-potentials. Further, we studied integration techniques to accelerate RL algorithms, and few methodologies to improve the practicality of the system.

In this section, we investigate issues or limitations with the presented contributions, and present avenues and directions for future work.

7.1 Unsupervised Detection of Eye-Blinks in EEG

We presented a fully automated and unsupervised eye-blink detection algorithm, *BLINK* [223] that self-learns user-specific brainwave profiles for eye-blinks. Following are the research directions to address some known issues with the *BLINK* algorithm, and towards designing a more robust algorithm.

7.1.1 Towards an online algorithm

BLINK algorithm is designed and presented as an offline algorithm. However, the *BLINK* algorithm can be used as-is in an online fashion. Real-time eye-blink detection widens the applicability of such an approach in the domain of BCI based communication, control, neurogaming, etc., and real-time EEG data processing. By design, the *BLINK* algorithm assumes the presence of a few (3+) similar eye-blinks in the EEG signal. Leveraging this fact, the proposed approach can be used as-is in an online manner by applying *BLINK* algorithm on a moving window with sufficient length ($\geq$ 30 seconds)[1].

We intend to extend this work by exploring the feasibility of optimizing this algorithm to operate in an online, real-time manner without using the moving window approach with repetitive computations. One of the directions to extend this approach is to dynamically build the correlation matrix and improve cluster formation upon peak detection during the continuous real-time monitoring of the EEG data. It is a challenging task to allow *BLINK* algorithm to detect eye-blinks without compromising the performance instantly, and can be studied in future work.

7.1.2 Other limitations of the *BLINK* algorithm

Despite the attractive performance score of *BLINK* algorithm, *BLINK* still fails to detect $\sim$50 eye-blink samples out of 2300 eye-blinks. We analyzed the undetected eye-blinks and concluded that failure cases, although being quite low ($<$2%), are mostly caused by the invalidity of the assumption of consistent eye-blink patterns within a subject. Occasionally,

[1]The average human eye-blink rate is 17 blinks/min in the rest condition [254]

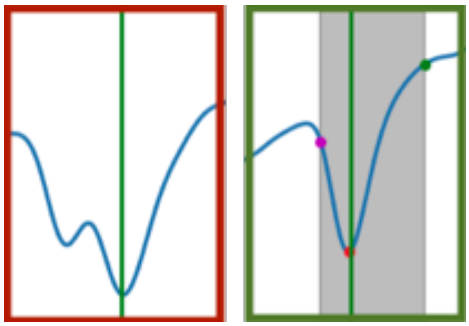


Figure 7.1: Failure cases of *BLINK* algorithm: (Left) abrupt eye-blink pattern not detected by *BLINK* algorithm, (Right) the regular eye-blink pattern exhibited by the user

an irregular eye-blink pattern was observed in the user data, which is quite dissimilar to the regular eye-blink pattern exhibited by the user. Figure 7.1 shows the cleaned irregular eye-blink pattern side by side with the regular eye-blink pattern. With the datasets collected in this work (Table 4.1), we plan to statistically evaluate the assumption of consistency in eye-blink patterns and improve the *BLINK* algorithm to consider such cases.

7.2 Lightweight EEG-based Wake-Up Command Design for BCI

We presented *Trance* [329], a user-friendly and robust wake-up command design and system for BCI headsets that is computationally lightweight, allowing BCI headsets to operate in a near-sleep mode but still reliably detect and interpret an EEG-based wake-up command from the user. We discuss in this subsection a few issues pertaining to *Trance* that are related, and can further improve the design

- ***STOP* command:** We only discussed the design of *START* command in our paper as by definition BCI platform is already in the active state, and thus recognition of *STOP* command need not be done necessarily under the constrained environment. The BCI cap running in the active mode, can either locally detect the command or "mobile-end" can perform the command detection with its massive computing capabilities and power availability, and ask the BCI cap to switch to low-power mode.

- **Timeout:** A timeout functionally is critical in systems using wake-up strategies as the user might forget that she issued the command earlier. With the limited energy availability at the BCI cap, the system should be smart to identify such cases and call a timeout to save energy.

- **Biofeedback:** A mechanism is necessary to notify the users if the issued command was correctly detected. In the event of successful detection of '*START* command at the BCI cap, it will start talking to the mobile-end, which in turn can be leveraged to provide the indication of an active state through vibrational notifications.

- **Compromised Visual Acuity:** Our framework was thoroughly tested on the healthy subjects without any discrimination of subjects who wear contact lenses or glasses, or visually impaired. Intuitively, subjects wearing glasses or lenses should not make a difference for the purpose of blink detection, but it would be interesting to see the implications of suffered vision on this framework, which we leave as future work.

- **Reinforcement-based accuracy improvement:** Once the user-issued command is successfully detected and the user initiates the normal BCI communication, it indicates the successful detection of the wake-up command. The *Trance* algorithm design can be improved to leverage this information (successful and false positive detection) to build an eye-blink fingerprint to further improve the accuracy rates.

7.2.1 Limitations of *Trance*

For the *EEG-MB* dataset, we analyzed the undetected wake-up commands, and concluded that the central cause of the failure case is the inconsistency of the consecutive blink patterns (as shown in Figure 7.2. We plan to (i) thoroughly evaluate such cases and improve *Trance* to account for them, and (ii) take an extra step to design efficient experimentation methodology to perform system testing in the wild, as a part of the future work. Moreover, once the user-issued command is successfully detected and the user initiates the normal BCI

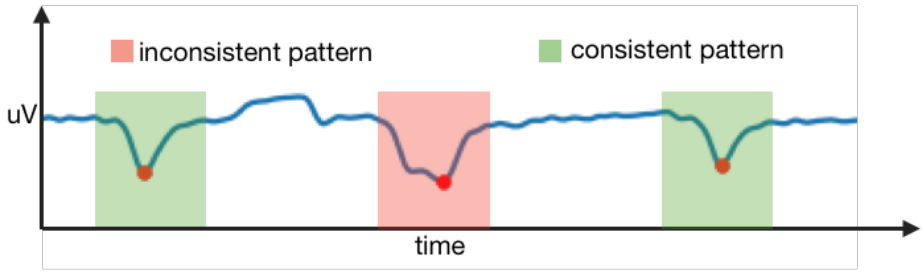


Figure 7.2: Inconsistency of blink patterns

communication, it indicates the successful detection of the wake-up command. *Trance* algorithm design can be improved to leverage this information (successful and false positive detection) to build an eye-blink fingerprint to improve the recall rates.

7.3 Tracking User Preferences using Brainwaves

We presented a machine learning algorithm *Cerebro* [330], which can learn the specific nuances of the user's brainwaves for preferences to accurately rank the objects. The analysis presented in [157], and in the presented work, is over grand average ERP waveforms obtained by averaging 50 waveforms. For the ERP analysis, it is a common practice to average multiple EEG waveforms to amplify the SNR, as the ERPs are phase-locked (synchronized) to the stimulus presentation. However, the strategy of averaging multiple waveforms is not practical from the usability standpoint, pertaining to the explicit requirements of passively watching the product multiple times. Such a system would be really inconvenient and unhandy to the users and would not be able to scale in the consumer market.

We perform a preliminary analysis to observe the effect of reducing the number of waveforms for an N200 feature based pairwise choice classification task (section 5.3.3, [157]). Figure 7.3 shows the accuracy per number of waveforms averaged for classification. We can see that by reducing the number of waveforms to half (i.e. 25), the accuracy drops only 5.4% (insignificant). This result though establishes the promise, demands an in-depth future work to move the needle close to the gold standard of one-waveform classification.

Further, this work can be extended in two main directions - (i) validating and eval-

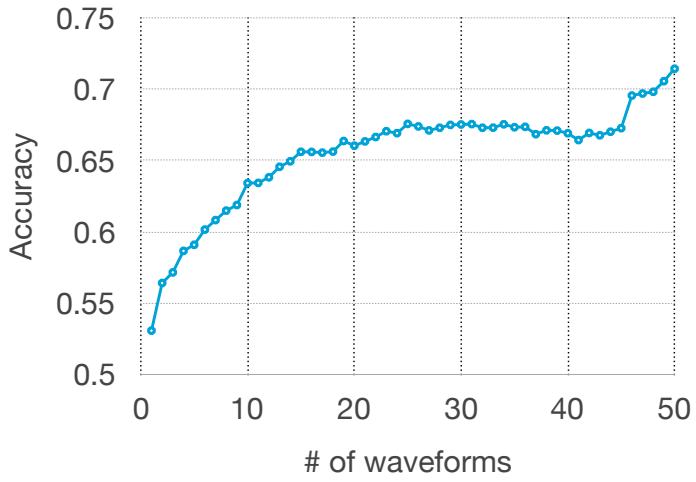


Figure 7.3: N200 accuracy with number of waveforms

uating *Cerebro* over a large corpus of user preference EEG data, and (ii) observing the performance of the algorithm in real-life conditions when the users are actually browsing products on their mobile devices.

7.4 On Using Brainwaves as Implicit Human Feedback in RL

We presented an interesting solution paradigm that will allow humans to assist ML algorithms (specifically, RL algorithms) without burdening human-in-the-loop through EEG-based brain waves [331, 332, 333]. Further directions to explore and advance the research in this domain are:

- Scalability to complex games and robotic environments: The scope of our work is limited to the visual-based RL problems with discrete state and action spaces. We have considered discrete grid-based reasonably complex navigational games in our work. Further studies have to be done to explore if such an approach could be extended to Atari and Robotic environments with very large state-space and continuous

action-space. This demands conducting human experiments over off-the-shelf Atari-games and real robotic environments and thus presents a multitude of system and synchronization challenges.

- ErrP generalizability considerations: The demonstration of the generalizability of error-potentials is also limited across the three reasonably complex environments presented in this work. We have considered discrete grid-based reasonably complex navigation games. The validation of the generalization to a variety of Atari and Robotic environments is the subject of future work. It'd be intriguing to explore the generalizability extent from simple to complex game environments, and the generalization capability of error-potentials between virtual and physical worlds. Further, an in-depth exploration of generalization study would be useful to investigate if error-potentials can be generalized over users, actions, etc.

- Multi-human ErrP: In the presented work, we have considered only single human based error-potential feedback to accelerate the convergence of the RL algorithm. Aggregating the individual ErrPs of multiple humans observing the same stimulation could lead to more reliability in error-potential detection, and hence the acceleration in convergence rate. Humans can have different physiological characteristics and different experimental conditions, such as the observation position. Hence, the reliability of ErrPs from different humans can vary significantly. As such, the ErrP detection accuracy can be improved by a weighted sum of ErrP detection results from multi-human EEG.

- Reward shaping considerations: We have considered two approaches to integrate the error-potential based human feedback with RL algorithms. Within reward shaping, we set a negative penalty as -0.75 on the detection of error-potentials. Could there be another optimal combination of augmented feedback leading to more acceleration which preserving the policy optimality? As part of the future work, non-trivial

approaches to reward shaping, or policy shaping with stochasticity in error-potential could be explored to improve the presented system.

- Hybrid fast learning with real-time human feedback: In the current scenario, we perform the human experiments prior to the training of the RL agent. Error-potential based human feedback is processed offline and stored in buckets, to be used during the training of the RL algorithms. In an ideal system, the agent plays the game at super-human speed (on GPUs), and in real-time presents the appropriate scenarios to the human observer to obtain the most meaningful feedback for learning. There remain several system-level frontiers to be solved, how to decouple the training and the human experimentation in real time, how frequently human users must be shown the recent trajectory of the agent, etc.

- Multi-modal feedback: In this work, we have considered only binary feedback (presence or absence) of error-potentials. The feedback can be made more information rich by considering other evoked potentials as implicit input including mismatches (N200), improbability (P300), semantic anomalies (N400), syntactic anomalies (P600), frequency of stimulus (SSEP), emotions, preferences (N200, ESRP), etc.

REFERENCES

[1] D. Saunders, "Dreams and esp," *Psi Encyclopedia. London: The Society for Psychical Research*, 2015.

[2] F. W. Myers, "Dreams and esp," *Psi Encyclopedia. London: The Society for Psychical Research*, 2017.

[3] Wikipedia contributors, *Mental radio — Wikipedia, the free encyclopedia*, 2016.

[4] H. Berger, "Über das elektrenkephalogramm des menschen," *European archives of psychiatry and clinical neuroscience*, vol. 87, no. 1, pp. 527–570, 1929.

[5] M. E. Menshawy, A. Benharref, and M. Serhani, "An automatic mobile-health based approach for eeg epileptic seizures detection," *Expert systems with applications*, vol. 42, no. 20, pp. 7157–7174, 2015.

[6] D.-M. Dobrea and M.-C. Dobrea, "An eeg (bio) technological system for assisting the disabled people," *2007 IEEE International Conference on Computational Cybernetics*, pp. 191–196, 2007.

[7] J. I. Ekandem, T. A. Davis, I. Alvarez, M. T. James, and J. E. Gilbert, "Evaluating the ergonomics of bci devices for research and experimentation," *Ergonomics*, vol. 55, no. 5, pp. 592–598, 2012.

[8] L.-D. Liao, C.-Y. Chen, I.-J. Wang, S.-F. Chen, S.-Y. Li, B.-W. Chen, J.-Y. Chang, and C.-T. Lin, "Gaming control using a wearable and wireless eeg-based brain-computer interface device with novel dry foam-based sensors," *Journal of neuroengineering and rehabilitation*, vol. 9, no. 1, p. 5, 2012.

[9] M.-K. Kang, H. Cho, H.-M. Park, S. C. Jun, and K.-J. Yoon, "A wellness platform for stereoscopic 3d video systems using eeg-based visual discomfort evaluation technology," *Applied ergonomics*, vol. 62, pp. 158–167, 2017.

[10] M. Bear, B. Connors, and M. A. Paradiso, *Neuroscience: Exploring the brain*. Jones & Bartlett Learning, LLC, 2020.

[11] J. F. Walvoord, *Daniel: The Key to Prophetic Revelation*. Moody Press, 1972.

[12] S. Somashekhar, M.-J. Sepúlveda, S Puglielli, A. Norden, E. Shortliffe, C Rohit Kumar, A Rauthan, N Arun Kumar, P Patil, K Rhee, *et al.*, "Watson for oncology and breast cancer treatment recommendations: Agreement with an expert multidisciplinary tumor board," *Annals of Oncology*, vol. 29, no. 2, pp. 418–423, 2018.

[13] W. Klimesch, "Eeg alpha and theta oscillations reflect cognitive and memory performance: A review and analysis," *Brain research reviews*, vol. 29, no. 2-3, pp. 169–195, 1999.

[14] P. Sauseng, W. Klimesch, M. Schabus, and M. Doppelmayr, "Fronto-parietal eeg coherence in theta and upper alpha reflect central executive functions of working memory," *International journal of Psychophysiology*, vol. 57, no. 2, pp. 97–103, 2005.

[15] S. Weiss and P. Rappelsberger, "Left frontal eeg coherence reflects modality independent language processes," *Brain Topography*, vol. 11, no. 1, pp. 33–42, 1998.

[16] E. Fonteneau, U. H. Frauenfelder, and L. Rizzi, "On the contribution of erps to the study of language comprehension," *Bulletin suisse de linguistique appliquée*, no. 68, pp. 111–124, 1998.

[17] M. D'Zmura, S. Deng, T. Lappas, S. Thorpe, and R. Srinivasan, "Toward eeg sensing of imagined speech," *Human-Computer Interaction. New Trends*, pp. 40–48, 2009.

[18] D. Girbau, "A neurocognitive approach to the study of private speech," *The Spanish Journal of Psychology*, vol. 10, no. 1, p. 41, 2007.

[19] V. Romei, V. Brodbeck, C. Michel, A. Amedi, A. Pascual-Leone, and G. Thut, "Spontaneous fluctuations in posterior α-band eeg activity reflect variability in excitability of human visual areas," *Cerebral cortex*, vol. 18, no. 9, pp. 2010–2018, 2008.

[20] T. A. Rihs, C. M. Michel, and G. Thut, "Mechanisms of selective inhibition in visual spatial attention are indexed by α-band eeg synchronization," *European Journal of Neuroscience*, vol. 25, no. 2, pp. 603–610, 2007.

[21] J. M. Kilner, Y. Paulignan, and D. Boussaoud, "Functional connectivity during real vs imagined visuomotor tasks: An eeg study," *Neuroreport*, vol. 15, no. 4, pp. 637–642, 2004.

[22] K. J. Miller, G. Schalk, E. E. Fetz, M. den Nijs, J. G. Ojemann, and R. P. Rao, "Cortical activity during motor execution, motor imagery, and imagery-based online feedback," *Proceedings of the National Academy of Sciences*, vol. 107, no. 9, pp. 4430–4435, 2010.

[23] M. Mikolajczak, K. Bodarwé, O. Laloyaux, M. Hansenne, and D. Nelis, "Association between frontal eeg asymmetries and emotional intelligence among adults," *Personality and Individual Differences*, vol. 48, no. 2, pp. 177–181, 2010.

[24] I. Papousek, H. H. Freudenthaler, and G. Schulter, "Typical performance measures of emotion regulation and emotion perception and frontal eeg asymmetry in an emotional contagion paradigm," *Personality and Individual Differences*, vol. 51, no. 8, pp. 1018–1022, 2011.

[25] G. Pfurtscheller, C. Brunner, A. Schlögl, and F. L. Da Silva, "Mu rhythm (de) synchronization and eeg single-trial classification of different motor imagery tasks," *NeuroImage*, vol. 31, no. 1, pp. 153–159, 2006.

[26] Y. R. Tabar and U. Halici, "A novel deep learning approach for classification of eeg motor imagery signals," *Journal of neural engineering*, vol. 14, no. 1, p. 016 003, 2016.

[27] A. N. Belkacem, D. Shin, H. Kambara, N. Yoshimura, and Y. Koike, "Online classification algorithm for eye-movement-based communication systems using two temporal eeg sensors," *Biomedical Signal Processing and Control*, vol. 16, pp. 40–47, 2015.

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

[29] D. Huang, C. Guan, K. K. Ang, H. Zhang, and Y. Pan, "Asymmetric spatial pattern for eeg-based emotion detection," *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, 2012.

[30] M Murugappan and S. Yaacob, "Asymmetric ratio and fcm based salient channel selection for human emotion detection using eeg," 2008.

[31] S. Deng, R. Srinivasan, T. Lappas, and M. D'Zmura, "Eeg classification of imagined syllable rhythm using hilbert spectrum methods," *Journal of neural engineering*, vol. 7, no. 4, p. 046 006, 2010.

[32] M. Matsumoto and J. Hori, "Classification of silent speech using support vector machine and relevance vector machine," *Applied Soft Computing*, vol. 20, pp. 95–102, 2014.

[33] A. P. Liavas, G. V. Moustakides, G. Henning, E. Z. Psarakis, and P. Husar, "A periodogram-based method for the detection of steady-state visually evoked potentials," *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 2, pp. 242–248, 1998.

[34] P. Wang, S. Ji-zhong, and S. Jin-he, "P300 detection algorithm based on fisher distance," *International Journal of Modern Education and Computer Science*, vol. 2, no. 2, p. 9, 2010.

[35] H. Cecotti and A. Graser, "Convolutional neural networks for p300 detection with application to brain-computer interfaces," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 3, pp. 433–445, 2010.

[36] J. Li, J. Liang, Q. Zhao, J. Li, K. Hong, and L. Zhang, "Design of assistive wheelchair system directly steered by human thoughts," *International journal of neural systems*, vol. 23, no. 03, p. 1 350 013, 2013.

[37] C. Neuper, G. R. Müller-Putz, R. Scherer, and G. Pfurtscheller, "Motor imagery and eeg-based control of spelling devices and neuroprostheses," *Progress in brain research*, vol. 159, pp. 393–409, 2006.

[38] C. Guan, M. Thulasidas, and J. Wu, "High performance p300 speller for brain-computer interface," *IEEE International Workshop on Biomedical Circuits and Systems, 2004.*, S3–5, 2004.

[39] M. De Vos, M. Kroesen, R. Emkes, and S. Debener, "P300 speller bci with a mobile eeg system: Comparison to a traditional amplifier," *Journal of neural engineering*, vol. 11, no. 3, p. 036 008, 2014.

[40] K. Brigham and B. V. Kumar, "Subject identification from electroencephalogram (eeg) signals during imagined speech," *2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pp. 1–8, 2010.

[41] M. A. Pisauro, E. Fouragnan, C. Retzler, and M. G. Philiastides, "Neural correlates of evidence accumulation during value-based decisions revealed via simultaneous eeg-fmri," *Nature communications*, vol. 8, no. 1, pp. 1–9, 2017.

[42] N. Goto, F. Mushtaq, D. Shee, X. L. Lim, M. Mortazavi, M. Watabe, and A. Schaefer, "Neural signals of selective attention are modulated by subjective preferences and buying decisions in a virtual shopping task," *Biological Psychology*, vol. 128, pp. 11–20, 2017.

[43] M. Calosio, E. Rybina, A. Shestakova, and V. Klucharev, "Transcranial direct current stimulation of the medial frontal cortex modulates choise-induced preference changes," *Higher School of Economics Research Paper No. WP BRP*, vol. 92, 2018.

[44] B. Kaneshiro, M. P. Guimaraes, H.-S. Kim, A. M. Norcia, and P. Suppes, "A representational similarity analysis of the dynamics of object processing using single-trial eeg classification," *Plos one*, vol. 10, no. 8, e0135697, 2015.

[45] A. Telpaz, R. Webb, and D. J. Levy, "Using eeg to predict consumers' future choices," *Journal of Marketing Research*, vol. 52, no. 4, pp. 511–529, 2015.

[46] C. Spampinato, S. Palazzo, I. Kavasidis, D. Giordano, N. Souly, and M. Shah, "Deep learning human mind for automated visual classification," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6809–6817, 2017.

[47] I. Kavasidis, S. Palazzo, C. Spampinato, D. Giordano, and M. Shah, "Brain2image: Converting brain signals into images," *Proceedings of the 25th ACM international conference on Multimedia*, pp. 1809–1817, 2017.

[48] A. F. Salazar-Gomez, J. DelPreto, S. Gil, F. H. Guenther, and D. Rus, "Correcting robot mistakes in real time using eeg signals," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6570–6577, 2017.

[49] R. Caton, "The electric currents of the brain," *Br Med J*, vol. 2, p. 278, 1875.

[50] V. Pravdich-Neminsky, "Experiments on the registration of the electrical phenomena of the mammalian brain," *Zbl. Physiol*, vol. 27, pp. 951–960, 1913.

[51] J. J. Vidal, "Toward direct brain-computer communication," *Annual review of Biophysics and Bioengineering*, vol. 2, no. 1, pp. 157–180, 1973.

[52] J. J. Vidal, "Real-time detection of brain events in eeg," *Proceedings of the IEEE*, vol. 65, no. 5, pp. 633–641, 1977.

[53] M. Velliste, S. Perel, M. C. Spalding, A. S. Whitford, and A. B. Schwartz, "Cortical control of a prosthetic arm for self-feeding," *Nature*, vol. 453, no. 7198, pp. 1098–1101, 2008.

[54] G. B. Stanley, F. F. Li, and Y. Dan, "Reconstruction of natural scenes from ensemble responses in the lateral geniculate nucleus," *The Journal of Neuroscience*, vol. 19, no. 18, pp. 8036–8042, 1999.

[55] M. A. Nicolelis, A. A. Ghazanfar, C. R. Stambaugh, L. M. Oliveira, M. Laubach, J. K. Chapin, R. J. Nelson, and J. H. Kaas, "Simultaneous encoding of tactile information by three primate cortical areas," *Nature neuroscience*, vol. 1, no. 7, pp. 621–630, 1998.

[56] J. Wessberg, C. R. Stambaugh, J. D. Kralik, P. D. Beck, M. Laubach, J. K. Chapin, J. Kim, S. J. Biggs, M. A. Srinivasan, and M. A. Nicolelis, "Real-time prediction of hand trajectory by ensembles of cortical neurons in primates," *Nature*, vol. 408, no. 6810, pp. 361–365, 2000.

[57] E. A. Pohlmeyer, E. R. Oby, E. J. Perreault, S. A. Solla, K. L. Kilgore, R. F. Kirsch, and L. E. Miller, "Toward the restoration of hand use to a paralyzed monkey: Brain-controlled functional electrical stimulation of forearm muscles," *PloS one*, vol. 4, no. 6, e5924, 2009.

[58] J. Naumann, *Search for Paradise: A Patient's Account of the Artificial Vision Experiment*. Xlibris Corporation, 2012.

[59] G. Pfurtscheller, G. R. Müller, J. Pfurtscheller, H. J. Gerner, and R. Rupp, " thought'–control of functional electrical stimulation to restore hand grasp in a patient with tetraplegia," *Neuroscience letters*, vol. 351, no. 1, pp. 33–36, 2003.

[60] G. Pfurtscheller, C Guger, G Müller, G Krausz, and C Neuper, "Brain oscillations control hand orthosis in a tetraplegic," *Neuroscience letters*, vol. 292, no. 3, pp. 211–214, 2000.

[61] G. R. Müller-Putz, R. Scherer, G. Pfurtscheller, and R. Rupp, "Eeg-based neuroprosthesis control: A step towards clinical practice," *Neuroscience letters*, vol. 382, no. 1, pp. 169–174, 2005.

[62] P. R. Kennedy and R. A. Bakay, "Restoration of neural output from a paralyzed patient by a direct brain connection," *Neuroreport*, vol. 9, no. 8, pp. 1707–1711, 1998.

[63] L. R. Hochberg, M. D. Serruya, G. M. Friehs, J. A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, and J. P. Donoghue, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature*, vol. 442, no. 7099, pp. 164–171, 2006.

[64] S. Smith, "Eeg in the diagnosis, classification, and management of patients with epilepsy," *Journal of Neurology, Neurosurgery & Psychiatry*, vol. 76, no. suppl 2, pp. ii2–ii7, 2005.

[65] A. Starr, "Auditory brain-stem responses in brain death.," *Brain: a journal of neurology*, vol. 99, no. 3, pp. 543–554, 1976.

[66] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain–computer interfaces for communication and control," *Clinical neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002.

[67] Y. Wang and T.-P. Jung, "A collaborative brain-computer interface for improving human performance," *PLoS One*, vol. 6, no. 5, e20422, 2011.

[68] A. Nijholt, "Bci for games: A 'state of the art'survey," *Entertainment Computing-ICEC 2008*, pp. 225–228, 2008.

[69] F Nijboer, B. Allison, S Dunne, D Plass-Oude Bos, A Nijholt, and P Haselager, "A preliminary survey on the perception of marketability of brain-computer interfaces and initial development of a repository of bci companies," 2011.

[70] J.-J. Vidal, "Toward direct brain-computer communication," *Annual review of Biophysics and Bioengineering*, vol. 2, no. 1, pp. 157–180, 1973.

[71] Vidal, Jacques J, "Real-time detection of brain events in eeg," *Proceedings of the IEEE*, vol. 65, no. 5, pp. 633–641, 1977.

[72] E. E. Sutter, "The brain response interface: Communication through visually-induced electrical brain responses," *Journal of Microcomputer Applications*, vol. 15, no. 1, pp. 31–45, 1992.

[73] M. Middendorf, G. McMillan, G. Calhoun, K. S. Jones, *et al.*, "Brain-computer interfaces based on the steady-state visual-evoked response," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 2, pp. 211–214, 2000.

[74] N Birbaumer, *Slow cortical potentials: Their origin, meaning, and clinical use*, 1997.

[75] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kübler, J. Perelmouter, E. Taub, and H. Flor, "A spelling device for the paralysed," *Nature*, vol. 398, no. 6725, pp. 297–298, 1999.

[76] B. Niels *et al.*, "The thought translation device (ttd) for completely paralyzed patients," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 2, 2000.

[77] A. Kübler, *Brain Computer Communication: Development of a Brain Computer Interface for Locked-in Patients on the Basis of the Psychophysiological Self-regulation Training of Slow Cortical Potentials (SCP)*. Schwäbische Verlags-Gesellschaft, 2000.

[78] E. Donchin, K. M. Spencer, and R. Wijesinghe, "The mental prosthesis: Assessing the speed of a p300-based brain-computer interface," *Rehabilitation Engineering, IEEE Transactions on*, vol. 8, no. 2, pp. 174–179, 2000.

[79] V. C. Blau, U. Maurer, N. Tottenham, and B. D. McCandliss, "The face-specific n170 component is modulated by emotional facial expression," *Behavioral and Brain Functions*, vol. 3, no. 1, p. 1, 2007.

[80] J. Perelmouter, B. Kotchoubey, A Kubler, E. Taub, and N. Birbaumer, "Language support program for thought-translation-devices," *Automedica*, vol. 18, no. 1, pp. 67–84, 1999.

[81] L. A. Farwell and E. Donchin, "Talking off the top of your head: Toward a mental prosthesis utilizing event-related brain potentials," *Electroencephalography and clinical Neurophysiology*, vol. 70, no. 6, pp. 510–523, 1988.

[82] P. Hagoort, "How the brain solves the binding problem for language: A neurocomputational model of syntactic processing," *Neuroimage*, vol. 20, S18–S29, 2003.

[83] M. van Vliet, C. Mühl, B. Reuderink, and M. Poel, "Guessing what's on your mind: Using the n400 in brain computer interfaces," *Brain Informatics*, pp. 180–191, 2010.

[84] G Pfurtscheller, C Brunner, A Schlögl, and F. L. Da Silva, "Mu rhythm (de) synchronization and eeg single-trial classification of different motor imagery tasks," *NeuroImage*, vol. 31, no. 1, pp. 153–159, 2006.

[85] D. McFarland, W. Sarnacki, T. Vaughan, and J. Wolpaw, "Eeg-based brain–computer interface communication effect of target number and trial length on information transfer rate," *Soc Neurosci Abstr 2000b*, vol. 26, p. 1228, 2000.

[86] B. Allison, R Leeb, C Brunner, G. Müller-Putz, G Bauernfeind, J. Kelly, and C Neuper, "Toward smarter bcis: Extending bcis through hybridization and intelligent control," *Journal of Neural Engineering*, vol. 9, no. 1, p. 013 001, 2011.

[87] J. A. Konstan, E. Chi, and K. Höök, "Proceedings of the sigchi conference on human factors in computing systems," 2012.

[88] A. Kapoor, P. Shenoy, and D. Tan, "Combining brain computer interfaces with vision for object categorization," *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, 2008.

[89] Q. Zhao, L. Zhang, and A. Cichocki, "Eeg-based asynchronous bci control of a car in 3d virtual reality environments," *Chinese Science Bulletin*, vol. 54, no. 1, pp. 78–87, 2009.

[90] *Mattel mindflex*, `http://service.mattel.com/us/productDetail.aspx?prodno=P2639&siteid=27`.

[91] *Neurosky mindwave*, `http://store.neurosky.com/products/mindwave-1`.

[92] *Neurosync*, `https://mindsolutionscorp.com/neurosync`.

[93] *Emotiv systems inc.* `https://emotiv.com/`.

[94] *Enobio*, `http://www.neuroelectrics.com/products/enobio/`.

[95] A. Delorme and S. Makeig, "Eeglab: An open source toolbox for analysis of single-trial eeg dynamics including independent component analysis," *Journal of neuroscience methods*, vol. 134, no. 1, pp. 9–21, 2004.

[96] Y. Renard, F. Lotte, G. Gibert, M. Congedo, E. Maby, V. Delannoy, O. Bertrand, and A. Lécuyer, "Openvibe: An open-source software platform to design, test, and use brain–computer interfaces in real and virtual environments," *Presence: teleoperators and virtual environments*, vol. 19, no. 1, pp. 35–53, 2010.

[97] L. Shoker, S. Sanei, and J. Chambers, "Artifact removal from electroencephalograms using a hybrid bss-svm algorithm," *IEEE Signal Processing Letters*, vol. 12, no. 10, pp. 721–724, 2005.

[98] B. Noureddin, P. D. Lawrence, and G. E. Birch, "Online removal of eye movement and blink eeg artifacts using a high-speed eye tracker," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 8, pp. 2103–2110, 2011.

[99] Y. Li, Z. Ma, W. Lu, and Y. Li, "Automatic removal of the eye blink artifact from eeg using an ica-based template matching approach," *Physiological measurement*, vol. 27, no. 4, p. 425, 2006.

[100] S. Hoffmann and M. Falkenstein, "The correction of eye blink artefacts in the eeg: A comparison of two prominent methods," *PLoS One*, vol. 3, no. 8, e3004, 2008.

[101] C. A. Joyce, I. F. Gorodnitsky, and M. Kutas, "Automatic removal of eye movement and blink artifacts from eeg data using blind component separation," *Psychophysiology*, vol. 41, no. 2, pp. 313–325, 2004.

[102] D.-h. Zhu, J.-j. Tong, and Y.-q. Chen, "An ica-based method for automatic eye blink artifact correction in multi-channel eeg," *Information Technology and Applications in Biomedicine, 2008. ITAB 2008. International Conference on*, pp. 338–341, 2008.

[103] T. Jung, C. Humphries, T.-W. Lee, S. Makeig, and M. McKeown, "Extended lca removes artifacts from electroencephalographic recordings," *Adv. Neural Inform. Process. Syst. 10*, vol. 10, pp. 894–900, 1998.

[104] F. C. Viola, J. Thorne, B. Edmonds, T. Schneider, T. Eichele, and S. Debener, "Semi-automatic identification of independent components representing eeg artifact," *Clinical Neurophysiology*, vol. 120, no. 5, pp. 868–877, 2009.

[105] N. Bigdely-Shamlo, K. Kreutz-Delgado, C. Kothe, and S. Makeig, "Eyecatch: Data-mining over half a million eeg independent components to construct a fully-automated eye-component detector," *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, pp. 5845–5848, 2013.

[106] H. Ghandeharion and A. Erfanian, "A fully automatic ocular artifact suppression from eeg data using higher order statistics: Improved performance by wavelet analysis," *Medical engineering & physics*, vol. 32, no. 7, pp. 720–729, 2010.

[107] Q. Zhao, B. Hu, Y. Shi, Y. Li, P. Moore, M. Sun, and H. Peng, "Automatic identification and removal of ocular artifacts in eeg—improved adaptive predictor filtering for portable applications," *IEEE transactions on nanobioscience*, vol. 13, no. 2, pp. 109–117, 2014.

[108] M. H. Soomro, N. Badruddin, M. Z. Yusoff, and M. A. Jatoi, "Automatic eye-blink artifact removal method based on emd-cca," *2013 ICME International Conference on Complex Medical Engineering*, pp. 186–190, 2013.

[109] P. He, G Wilson, and C Russell, "Removal of ocular artifacts from electro-encephalogram by adaptive filtering," *Medical and biological engineering and computing*, vol. 42, no. 3, pp. 407–412, 2004.

[110] S. M. Haas, M. G. Frei, I. Osorio, B. Pasik-Duncan, and J. Radel, "Eeg ocular artifact removal through armax model system identification using extended least squares," *Communications in information and systems*, vol. 3, no. 1, pp. 19–40, 2003.

[111] J. Mateo, A. M. Torres, and M. A. García, "Eye interference reduction in electroencephalogram recordings using a radial basic function," *IET Signal Processing*, vol. 7, no. 7, pp. 565–576, 2013.

[112] H. Shin, H. Kim, S. Lee, and J. Kang, "Online removal of ocular artifacts from single channel eeg for ubiquitous healthcare applications," *Proceedings of the 4th International Conference on Ubiquitous Information Technologies & Applications*, pp. 1–6, 2009.

[113] C. A. Majmudar, R. Mahajan, and B. I. Morshed, "Real-time hybrid ocular artifact detection and removal for single channel eeg," *2015 IEEE International Conference on Electro/Information Technology (EIT)*, pp. 330–334, 2015.

[114] S. Sreeja, R. R. Sahay, D. Samanta, and P. Mitra, "Removal of eye blink artifacts from eeg signals using sparsity," *IEEE journal of biomedical and health informatics*, vol. 22, no. 5, pp. 1362–1372, 2017.

[115] S. Zhang, J. McIntosh, S. M. Shadli, P. S. Neo, Z. Huang, and N. McNaughton, "Removing eye blink artefacts from eeg—a single-channel physiology-based method," *Journal of neuroscience methods*, vol. 291, pp. 213–220, 2017.

[116] H. Nolan, R. Whelan, and R. B. Reilly, "Faster: Fully automated statistical thresholding for eeg artifact rejection," *Journal of neuroscience methods*, vol. 192, no. 1, pp. 152–162, 2010.

[117] A. Klein and W. Skrandies, "A reliable statistical method to detect eyeblink-artefacts from electroencephalogram data only," *Brain topography*, vol. 26, no. 4, pp. 558–568, 2013.

[118] A. Egambaram, N. Badruddin, V. S. Asirvadam, E. Fauvet, C. Stolz, and T. Begum, "Unsupervised eye blink artifact identification in electroencephalogram," *TENCON 2018-2018 IEEE Region 10 Conference*, pp. 2148–2152, 2018.

[119] W.-D. Chang, H.-S. Cha, K. Kim, and C.-H. Im, "Detection of eye blink artifacts from single prefrontal channel electroencephalogram," *Computer methods and programs in biomedicine*, vol. 124, pp. 19–30, 2016.

[120] S. Maruthachalam, M. Kumar, and H. Murthy, "Time warping solutions for classifying artifacts in eeg," Jul. 2019.

[121] M. bin Abd Rani *et al.*, "Detection of eye blinks from eeg signals for home lighting system activation," in *2009 6th International Symposium on Mechatronics and its Applications*, 2009.

[122] W.-D. Chang and C.-H. Im, "Enhanced template matching using dynamic positional warping for identification of specific patterns in electroencephalogram," *Journal of Applied Mathematics*, vol. 2014, 2014.

[123] R. Ghosh, N. Sinha, and S. K. Biswas, "Automated eye blink artefact removal from eeg using support vector machine and autoencoder," *IET Signal Processing*, vol. 13, no. 2, pp. 141–148, 2018.

[124] S Rihana, P Damien, and T Moujaess, "Eeg-eye blink detection system for brain computer interface," *Converging Clinical and Engineering Research on Neurorehabilitation*, pp. 603–608, 2013.

[125] R. Paprocki, T. Gebrehiwot, M. Gradinscak, and A. Lenskiy, "Extracting blink rate variability from eeg signals," *arXiv preprint arXiv:1603.03031*, 2016.

[126] Z. Tiganj, M. Mboup, C. Pouzat, and L. Belkoura, "An algebraic method for eye blink artifacts detection in single channel eeg recordings," *17th International Conference on Biomagnetism Advances in Biomagnetism–Biomag2010*, pp. 175–178, 2010.

[127] M. Rohál'ová, P. Sykacek, M Koskaand, and G. Dorffner, "Detection of the eeg artifacts by the means of the (extended) kalman filter," *Meas. Sci. Rev*, vol. 1, no. 1, pp. 59–62, 2001.

[128] O. Dehzangi, A. Melville, and M. Taherisadr, "Automatic eeg blink detection using dynamic time warping score clustering," *Advances in Body Area Networks I*, pp. 49–60, 2019.

[129] G. Gratton, M. G. Coles, and E. Donchin, "A new method for off-line removal of ocular artifact," *Electroencephalography and clinical neurophysiology*, vol. 55, no. 4, pp. 468–484, 1983.

[130] R. J. Croft and R. J. Barry, "Eog correction: Which regression should we use?" *Psychophysiology*, vol. 37, no. 1, pp. 123–125, 2000.

[131] R. J. Croft and R. J. Barry, "Removal of ocular artifact from the eeg: A review," *Neurophysiologie Clinique/Clinical Neurophysiology*, vol. 30, no. 1, pp. 5–19, 2000.

[132] T. Kim, S. Hwang, S. Kim, H. Ahn, and D. Chung, *Smart contact lenses for augmented reality and methods of manufacturing and operating the same*, US Patent App. 14/644,488, 2016.

[133] P. M. Corcoran, F. Nanu, S. Petrescu, and P. Bigioi, "Real-time eye gaze tracking for gaming design and consumer electronics systems," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 2, 2012.

[134] A. J. van Breemen, "Animation engine for believable interactive user-interface robots," *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 3, pp. 2873–2878, 2004.

[135] A. E. Kaufman, A. Bandopadhay, and B. D. Shaviv, "An eye tracking computer user interface," *Virtual Reality, 1993. Proceedings., IEEE 1993 Symposium on Research Frontiers in*, pp. 120–121, 1993.

[136] E. Gupta, M. Agarwal, and R. Sivakumar, "Blink to get in: Biometric authentication for mobile devices using eeg signals," *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2020.

[137] B. Noronha, S. Dziemian, G. A. Zito, C. Konnaris, and A. A. Faisal, "Wink to grasp—comparing eye, voice & emg gesture control of grasp with soft-robotic gloves," *2017 International Conference on Rehabilitation Robotics (ICORR)*, pp. 1043–1048, 2017.

[138] W. W. Abbott and A. A. Faisal, "Ultra-low-cost 3d gaze estimation: An intuitive high information throughput compliment to direct brain–machine interfaces," *Journal of neural engineering*, vol. 9, no. 4, p. 046 016, 2012.

[139] R. O. Maimon-Mor, J. Fernandez-Quesada, G. A. Zito, C. Konnaris, S. Dziemian, and A. A. Faisal, "Towards free 3d end-point control for robotic-assisted human reaching using binocular eye tracking," *2017 International Conference on Rehabilitation Robotics (ICORR)*, pp. 1049–1054, 2017.

[140] K. Grauman, M. Betke, J. Gips, and G. R. Bradski, "Communication via eye blinks-detection and duration analysis in real time," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, pp. I–I, 2001.

[141] C. H. Morimoto and M. R. Mimica, "Eye gaze tracking techniques for interactive applications," *Computer vision and image understanding*, vol. 98, no. 1, pp. 4–24, 2005.

[142] B. Tag, J. Shimizu, C. Zhang, N. Ohta, K. Kunze, and K. Sugiura, "Eye blink as an input modality for a responsive adaptable video system," *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, pp. 205–208, 2016.

[143] M. Pike, R. Ramchurn, S. Benford, and M. L. Wilson, "# scanners: Exploring the control of adaptive films using brain-computer interaction," *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 5385–5396, 2016.

[144] M. X. Huang, T. C. Kwok, G. Ngai, S. C. Chan, and H. V. Leong, "Building a personalized, auto-calibrating eye tracker from user interactions," *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 5169–5179, 2016.

[145] I. Chatterjee, R. Xiao, and C. Harrison, "Gaze+ gesture: Expressive, precise and targeted free-space interactions," *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pp. 131–138, 2015.

[146] Q. Ma, X. Wang, S. Dai, and L. Shu, "Event-related potential n270 correlates of brand extension," *Neuroreport*, vol. 18, no. 10, pp. 1031–1034, 2007.

[147] Q. Ma, X. Wang, L. Shu, and S. Dai, "P300 and categorization in brand extension," *Neuroscience letters*, vol. 431, no. 1, pp. 57–61, 2008.

[148] X. Wang, Q. Ma, and C. Wang, "N400 as an index of uncontrolled categorization processing in brand extension," *Neuroscience letters*, vol. 525, no. 1, pp. 76–81, 2012.

[149] M Murugappan, S. Murugappan, C. Gerard, *et al.*, "Wireless eeg signals based neuromarketing system using fast fourier transform (fft)," *2014 IEEE 10th International Colloquium on Signal Processing and its Applications*, pp. 25–30, 2014.

[150] R. Pozharliev, W. J. Verbeke, J. W. Van Strien, and R. P. Bagozzi, "Merely being with you increases my attention to luxury products: Using eeg to understand consumers' emotional experience with luxury branded products," *Journal of Marketing Research*, vol. 52, no. 4, pp. 546–558, 2015.

[151] M.-H. Lin, S. N. Cross, and T. L. Childers, "Understanding olfaction and emotions and the moderating role of individual differences," *European Journal of Marketing*, vol. 52, no. 3/4, pp. 811–836, 2018.

[152] M. Chen, Q. Ma, M. Li, S. Dai, X. Wang, and L. Shu, "The neural and psychological basis of herding in purchasing books online: An event-related potential study," *Cyberpsychology, Behavior, and Social Networking*, vol. 13, no. 3, pp. 321–328, 2010.

[153] F. Guo, X. Zhang, Y. Ding, and X. Wang, "Recommendation influence: Differential neural responses of consumers during shopping online.," *Journal of Neuroscience, Psychology, and Economics*, vol. 9, no. 1, p. 29, 2016.

[154] W. J. Jones, T. L. Childers, and Y. Jiang, "The shopping brain: Math anxiety modulates brain responses to buying decisions," *Biological Psychology*, vol. 89, no. 1, pp. 201–213, 2012.

[155] R. N. Khushaba, L. Greenacre, S. Kodagoda, J. Louviere, S. Burke, and G. Dissanayake, "Choice modeling and the brain: A study on the electroencephalogram (eeg) of preferences," *Expert Systems with Applications*, vol. 39, no. 16, pp. 12378 –12 388, 2012.

[156] N. Goto, F. Mushtaq, D. Shee, X. L. Lim, M. Mortazavi, M. Watabe, and A. Schaefer, "Neural signals of selective attention are modulated by subjective preferences and buying decisions in a virtual shopping task," *Biological Psychology*, vol. 128, pp. 11 –20, 2017.

[157] A. Telpaz, R. Webb, and D. J. Levy, "Using eeg to predict consumers' future choices," *Journal of Marketing Research*, vol. 52, no. 4, pp. 511–529, 2015.

[158] D. Baldo, H. Parikh, Y. Piu, and K.-M. Müller, "Brain waves predict success of new fashion products: A practical application for the footwear retailing industry," *Journal of Creating Value*, vol. 1, no. 1, pp. 61–71, 2015.

[159] M. Yadava, P. Kumar, R. Saini, P. P. Roy, and D. Prosad Dogra, "Analysis of eeg signals and its application to neuromarketing," *Multimedia Tools and Applications*, vol. 76, no. 18, pp. 19 087–19 111, Sep. 2017.

[160] C. S. Carter, T. S. Braver, D. M. Barch, M. M. Botvinick, D. Noll, and J. D. Cohen, "Anterior cingulate cortex, error detection, and the online monitoring of performance," *Science*, vol. 280, no. 5364, pp. 747–749, 1998.

[161] P. W. Ferrez and J. d. R. Millán, "You are wrong!—automatic detection of interaction errors from brain waves," in *Proceedings of the 19th international joint conference on Artificial intelligence*, 2005.

[162] M. Falkenstein, J. Hoormann, S. Christ, and J. Hohnsbein, "Erp components on reaction errors and their functional significance: A tutorial," *Biological psychology*, vol. 51, no. 2-3, pp. 87–107, 2000.

[163] B. Dal Seno, M. Matteucci, and L. Mainardi, "Online detection of p300 and error potentials in a bci speller," *Computational intelligence and neuroscience*, vol. 2010, 2010.

[164] I. Iturrate, L. Montesano, and J. Minguez, "Robot reinforcement learning using eeg-based reward signals," *2010 IEEE International Conference on Robotics and Automation*, pp. 4822–4829, 2010.

[165] M. Spüler and C. Niethammer, "Error-related potentials during continuous feedback: Using eeg to detect errors of different type and severity," *Frontiers in human neuroscience*, vol. 9, p. 155, 2015.

[166] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Multiclass brain–computer interface classification by riemannian geometry," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 4, pp. 920–928, 2011.

[167] M. Congedo, A. Barachant, and A. Andreev, "A new generation of brain-computer interface based on riemannian geometry," *arXiv preprint arXiv:1310.8115*, 2013.

[168] C. Daniel, O. Kroemer, M. Viering, J. Metz, and J. Peters, "Active reward learning with a novel acquisition function," *Autonomous Robots*, vol. 39, no. 3, pp. 389–405, 2015.

[169] L. El Asri, B. Piot, M. Geist, R. Laroche, and O. Pietquin, "Score-based inverse reinforcement learning," *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pp. 457–465, 2016.

[170] S. I. Wang, P. Liang, and C. D. Manning, "Learning language games through interaction," *arXiv preprint arXiv:1606.02447*, 2016.

[171] C. Wirth, J. Fürnkranz, and G. Neumann, "Model-free preference-based reinforcement learning," *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[172] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems*, vol. 30, pp. 4299–4307, 2017.

[173] C. B. Holroyd, S. Nieuwenhuis, N. Yeung, and J. D. Cohen, "Errors in reward prediction are reflected in the event-related brain potential," *Neuroreport*, vol. 14, no. 18, pp. 2481–2484, 2003.

[174] C. B. Holroyd and M. G. Coles, "The neural basis of human error processing: Reinforcement learning, dopamine, and the error-related negativity.," *Psychological review*, vol. 109, no. 4, p. 679, 2002.

[175] G. Schalk, J. R. Wolpaw, D. J. McFarland, and G. Pfurtscheller, "Eeg-based communication: Presence of an error potential," *Clinical neurophysiology*, vol. 111, no. 12, pp. 2138–2144, 2000.

[176] B. Blankertz, G. Dornhege, C. Schafer, R. Krepki, J. Kohlmorgen, K.-R. Muller, V. Kunzmann, F. Losch, and G. Curio, "Boosting bit rates and error detection for the classification of fast-paced motor commands based on single-trial eeg analysis," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 11, no. 2, pp. 127–131, 2003.

[177] L. C. Parra, C. D. Spence, A. D. Gerson, and P. Sajda, "Response error correction-a demonstration of improved human-machine performance using real-time eeg monitoring," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 11, no. 2, pp. 173–177, 2003.

[178] P. W. Ferrez and J. d. R. Millán, "Error-related eeg potentials generated during simulated brain–computer interaction," *IEEE transactions on biomedical engineering*, vol. 55, no. 3, pp. 923–929, 2008.

[179] S. K. Kim, E. A. Kirchner, A. Stefes, and F. Kirchner, "Intrinsic interactive reinforcement learning–using error-related potentials for real world human-robot interaction," *Scientific reports*, vol. 7, no. 1, p. 17 562, 2017.

[180] R. Chavarriaga and J. d. R. Millán, "Learning from eeg error-related potentials in noninvasive brain-computer interfaces," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 18, no. 4, pp. 381–388, 2010.

[181] P. Berg and M. Scherg, "Dipole models of eye movements and blinks," *Electroencephalography and clinical neurophysiology*, vol. 79, no. 1, pp. 36–44, 1991.

[182] P. Berg, "The residual after correcting event-related potentials for blink artifacts," *Psychophysiology*, vol. 23, no. 3, pp. 354–364, 1986.

[183] P. Berg and M. Scherg, "Dipole models of eye movements and blinks," *Electroencephalography and clinical Neurophysiology*, vol. 79, no. 1, pp. 36–44, 1991.

[184] P. Berg, "The residual after correcting event-related potentials for blink artifacts," *Psychophysiology*, vol. 23, no. 3, pp. 354–364, 1986.

[185] D. Nelligan, "Eye movement artifacts and electrical recording of eye position," *Proc. EPTA*, vol. 11, pp. 25–43, 1964.

[186] T. Robinson and E. Johnson, "Eye movement artifact as an example of volume conduction," *Proceedings of the Sixth International Congress. Electroencephalogr Clin Neurophysiol*, pp. 565–7, 1965.

[187] E. Marg, "Development of electro-oculography: Standing potential of the eye in registration of eye movement," *AMA archives of ophthalmology*, vol. 45, no. 2, pp. 169–185, 1951.

[188] W. Miles, "Eyeball reflex movement associated with voluntary and reflex winking," *Am J Physiol*, vol. 72, p. 239, 1925.

[189] S. Newhall and H. Halverson, "Eye-movements correlated with innervation of the orbicularis oculi," *The Journal of General Psychology*, vol. 11, no. 2, pp. 287–300, 1934.

[190] R. W. Lawson, *Blinking: Its role in physical measurements*, 1948.

[191] W. Blount, "Studies of the movements of the eyelids of animals: Blinking," *Quarterly Journal of Experimental Physiology: Translation and Integration*, vol. 18, no. 2, pp. 111–125, 1927.

[192] F. Matsuo, J. F. Peters, and E. L. Reilly, "Electrical phenomena associated with movements of the eyelid," *Electroencephalography and clinical neurophysiology*, vol. 38, no. 5, pp. 507–511, 1975.

[193] W Barry and G. M. Jones, "Influence of eye lid movement upon electro-oculographic recording of vertical eye movements.," *Aerospace medicine*, vol. 36, p. 855, 1965.

[194] *Biopac cap-100c*, `https://www.biopac.com/product/eeg-caps-for-cap100c/`, 2019.

[195] *Interaxon muse headband*, `http://www.choosemuse.com`, 2019.

[196] *Muse monitor application*, `https://musemonitor.com/`, 2019.

[197] *Automated eye blink detection online*, `http://openbci.com/community/automated-eye-blink-detection-online-2/`, 2019.

[198] S. Hoffmann and M. Falkenstein, "The correction of eye blink artefacts in the eeg: A comparison of two prominent methods," *PLoS One*, vol. 3, no. 8, e3004, 2008.

[199] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[200] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[201] S. Pinker, "The brain: The mystery of consciousness," *Time Magazine*, 2007.

[202] K Pribram, "Reflections on the place of brain in the ecology of mind," *Cognition and the symbolic processes*, vol. 2, 1980.

[203] A. Puri, "Acceptance and usage of smart wearable devices in canadian older adults," M.S. thesis, University of Waterloo, 2017.

[204] A. Adapa, F. F.-H. Nah, R. H. Hall, K. Siau, and S. N. Smith, "Factors influencing the adoption of smart wearable devices," *International Journal of Human–Computer Interaction*, vol. 34, no. 5, pp. 399–409, 2018.

[205] A. Kononova, L. Li, K. Kamp, M. Bowen, R. Rikard, S. Cotten, and W. Peng, "The use of wearable activity trackers among older adults: Focus group study of tracker perceptions, motivators, and barriers in the maintenance stage of behavior change," *JMIR mHealth and uHealth*, vol. 7, no. 4, e9832, 2019.

[206] D. Pal, S. Funilkul, and V. Vanijja, "The future of smartwatches: Assessing the end-users' continuous usage using an extended expectation-confirmation model," *Universal Access in the Information Society*, pp. 1–21, 2018.

[207] C. Maher, J. Ryan, C. Ambrosi, and S. Edney, "Users' experiences of wearable activity trackers: A cross-sectional study," *BMC public health*, vol. 17, no. 1, p. 880, 2017.

[208] Emotiv, *Emotiv epoc+ battery life*, https://www.emotiv.com/epoc/, 2019.

[209] Emotiv, *Emotiv insight battery life*, https://www.emotiv.com/product/emotiv-insight-5-channel-mobile-eeg/tab-description, 2019.

[210] Muse, *Muse battery life*, https://neurobb.com/t/muse-battery-replacement/665, 2019.

[211] A. Insights, *Battery life still important to wearable consumers*, http://www.argusinsights.com/2016/04/14/battery-life-still-important-to-wearable-consumers/, Apr. 2016.

[212] B. Reed, *Battery life has become the single biggest reason people choose a smartphone*, http://bgr.com/2014/05/12/best-smartphone-battery-life/, Mar. 2014.

[213] P. Pickering, *The importance of battery technology in wearables*, https://www.ecnmag.com/article/2015/09/importance-battery-technology-wearables, Sep. 2014.

[214] J. P. Powell, *Display brightness control method and apparatus for conserving battery power*, US Patent 6,618,042, Sep. 2003.

[215] Embedded, *Optimizing wearable display power consumption*, https://www.embedded.com/design/power-optimization/4461973/Optimizing-wearable-display-power-consumption, 2019.

[216] J. P. Karidis and C. A. Pickover, *Apparatus and method for display power saving*, US Patent 7,614,011, Nov. 2009.

[217] X. Liu, T. Chen, F. Qian, Z. Guo, F. X. Lin, X. Wang, and K. Chen, "Characterizing smartwatch usage in the wild," *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 385–398, 2017.

[218] G. L. Calhoun and G. R. McMillan, "Hands-free input devices for wearable computers," *Proceedings Fourth Annual Symposium on Human Interaction with Complex Systems*, pp. 118–123, 1998.

[219] S. Schaffer, R. Schleicher, and S. Möller, "Modeling input modality choice in mobile graphical and speech interfaces," *International Journal of Human-Computer Studies*, vol. 75, pp. 21–34, 2015.

[220] T. Simpson, C. Broughton, M. J. Gauthier, and A. Prochazka, "Tooth-click control of a hands-free computer interface," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 8, pp. 2050–2056, 2008.

[221] I. Wechsung, K.-P. Engelbrecht, and S. Möller, "Using quality ratings to predict modality choice in multimodal systems," *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[222] A. L. Rudnicky, "Mode preference in a simple data-retrieval task," in *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*, 1993.

[223] M. Agarwal and R. Sivakumar, "Blink: A fully automated unsupervised algorithm for eye-blink detection in eeg signals," in *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, IEEE, 2019.

[224] S. Zhao, P. Dragicevic, M. Chignell, R. Balakrishnan, and P. Baudisch, "Earpod: Eyes-free menu selection using touch input and reactive audio feedback," *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 1395–1404, 2007.

[225] J. Schwarz, C. Harrison, S. Hudson, and J. Mankoff, "Cord input: An intuitive, high-accuracy, multi-degree-of-freedom input method for mobile devices," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1657–1660, 2010.

[226] P. Baudisch and G. Chu, "Back-of-device interaction allows creating very small touch devices," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1923–1932, 2009.

[227] G. Blasko and S. Feiner, "An interaction system for watch computers using tactile guidance and bidirectional segmented strokes," *Eighth International Symposium on Wearable Computers*, vol. 1, pp. 120–123, 2004.

[228] M. M. Punt, C. N. Stefels, C. A. Grimbergen, and J. Dankelman, "Evaluation of voice control, touch panel control and assistant control during steering of an endoscope," *Minimally Invasive Therapy & Allied Technologies*, vol. 14, no. 3, pp. 181–187, 2005.

[229] C. S. L. Tsui, P. Jia, J. Q. Gan, H. Hu, and K. Yuan, "Emg-based hands-free wheelchair control with eog attention shift detection," *2007 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1266–1271, 2007.

[230] T. Felzer, R. Fischer, T. Groensfelder, and R. Nordmann, "Alternative control system for operating a pc using intentional muscle contractions only," *Online-Proc. CSUN Conf*, 2005.

[231] J. F. Hipp and M. Siegel, "Dissociating neuronal gamma-band activity from cranial and ocular muscle activity in eeg," *Frontiers in human neuroscience*, vol. 7, p. 338, 2013.

[232] T. S. Saponas, D. S. Tan, D. Morris, R. Balakrishnan, J. Turner, and J. A. Landay, "Enabling always-available input with muscle-computer interfaces," *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pp. 167–176, 2009.

[233] A. Soares, A. Andrade, E. Lamounier, and R. Carrijo, "The development of a virtual myoelectric prosthesis controlled by an emg pattern recognition system based on neural networks," *Journal of Intelligent Information Systems*, vol. 21, no. 2, pp. 127–141, 2003.

[234] S. J. Scrivani, D. A. Keith, and L. B. Kaban, "Temporomandibular disorders," *New England Journal of Medicine*, vol. 359, no. 25, pp. 2693–2705, 2008.

[235] J. Kela, P. Korpipää, J. Mäntyjärvi, S. Kallio, G. Savino, L. Jozzo, and D. Marca, "Accelerometer-based gesture control for a design environment," *Personal and Ubiquitous Computing*, vol. 10, no. 5, pp. 285–299, 2006.

[236] V. Këpuska and T. Klein, "A novel wake-up-word speech recognition system, wake-up-word recognition task, technology and evaluation," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 71, no. 12, e2772–e2789, 2009.

[237] O. Novanda, M. Salem, J. Saunders, M. L. Walters, and K. Dautenhahn, "What communication modalities do users prefer in real time hri?" *arXiv preprint arXiv:1606.03992*, 2016.

[238] M. Agarwal and R. Sivakumar, "Think: Toward practical general-purpose brain-computer communication," *Proceedings of the 2Nd International Workshop on Hot Topics in Wireless*, pp. 41–45, 2015.

[239] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, and B. Arnaldi, "A review of classification algorithms for eeg-based brain–computer interfaces," *Journal of neural engineering*, vol. 4, no. 2, R1, 2007.

[240] F. Lotte, L. Bougrain, A. Cichocki, M. Clerc, M. Congedo, A. Rakotomamonjy, and F. Yger, "A review of classification algorithms for eeg-based brain–computer interfaces: A 10 year update," *Journal of neural engineering*, vol. 15, no. 3, p. 031 005, 2018.

[241] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain–computer interfaces for communication and control," *Clinical neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002.

[242] M. Middendorf, G. McMillan, G. Calhoun, and K. S. Jones, "Brain-computer interfaces based on the steady-state visual-evoked response," *IEEE transactions on rehabilitation engineering*, vol. 8, no. 2, pp. 211–214, 2000.

[243] G. Townsend, B. Graimann, and G. Pfurtscheller, "Continuous eeg classification during motor imagery-simulation of an asynchronous bci," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 12, no. 2, pp. 258–265, 2004.

[244] D. Coyle, J. Garcia, A. R. Satti, and T. M. McGinnity, "Eeg-based continuous control of a game using a 3 channel motor imagery bci: Bci game," *Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB), 2011 IEEE Symposium on*, pp. 1–7, 2011.

[245] P.-J. Kindermans, H. Verschore, D. Verstraeten, and B. Schrauwen, "A p300 bci for the masses: Prior information enables instant unsupervised spelling," *Advances in Neural Information Processing Systems*, pp. 710–718, 2012.

[246] R. C. Panicker, S. Puthusserypady, and Y. Sun, "An asynchronous p300 bci with ssvep-based control state detection," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 6, pp. 1781–1788, 2011.

[247] G. Bin, X. Gao, Y. Wang, B. Hong, and S. Gao, "Vep-based brain-computer interfaces: Time, frequency, and code modulations [research frontier]," *IEEE Computational Intelligence Magazine*, vol. 4, no. 4, 2009.

[248] G. Bin, X. Gao, Y. Wang, Y. Li, B. Hong, and S. Gao, "A high-speed bci based on code modulation vep," *Journal of neural engineering*, vol. 8, no. 2, p. 025 015, 2011.

[249] S. Inoue, Y. Akiyama, Y. Izumi, and S. Nishijima, "The development of bci using alpha waves for controlling the robot arm," *IEICE transactions on communications*, vol. 91, no. 7, pp. 2125–2132, 2008.

[250] P. Batres-Mendoza, M. A. Ibarra-Manzano, E. I. Guerra-Hernandez, D. L. Almanza-Ojeda, C. R. Montoro-Sanjose, R. J. Romero-Troncoso, and H. Rostro-Gonzalez,

"Improving eeg-based motor imagery classification for real-time applications using the qsa method," *Computational intelligence and neuroscience*, vol. 2017, 2017.

[251] N. Padfield, J. Zabalza, H. Zhao, V. Masero, and J. Ren, "Eeg-based brain-computer interfaces using motor-imagery: Techniques and challenges," *Sensors*, vol. 19, no. 6, p. 1423, 2019.

[252] J. Kevric and A. Subasi, "Comparison of signal decomposition methods in classification of eeg signals for motor-imagery bci system," *Biomedical Signal Processing and Control*, vol. 31, pp. 398–406, 2017.

[253] K. Nakamori, M. Odawara, T. Nakajima, T. Mizutani, and K. Tsubota, "Blinking is controlled primarily by ocular surface conditions," *American journal of ophthalmology*, vol. 124, no. 1, pp. 24–30, 1997.

[254] A. R. Bentivoglio, S. B. Bressman, E. Cassetta, D. Carretta, P. Tonali, and A. Albanese, "Analysis of blink rate patterns in normal subjects," *Movement Disorders*, vol. 12, no. 6, pp. 1028–1034, 1997.

[255] M. Agarwal and R. Sivakumar, "Poster: Characters vs. words: Observations on command design for brain-computer interfaces," *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 177–177, 2017.

[256] J. Pan and W. J. Tompkins, "A real-time qrs detection algorithm," *IEEE Trans. Biomed. Eng*, vol. 32, no. 3, pp. 230–236, 1985.

[257] Vocalize.ai, *Smart speakers, wake words and ghoul oil*, `http://www.vocalize.ai/2018/11/01/wake-words-false-positives/`, 1998.

[258] Picovoice, *Wake word engine benchmark frameworks*, `https://github.com/Picovoice/wake-word-benchmark`, 2019.

[259] B. Farnsworth, *Eeg headset prices – an overview of 15+ eeg devices*, `https://imotions.com/blog/eeg-headset-prices/`, 2019.

[260] A. Vourvopoulos *et al.*, "Usability and cost-effectiveness in brain-computer interaction: Is it user throughput or technology related?" *Proceedings of the 7th Augmented Human International Conference 2016*, p. 19, 2016.

[261] A. Aldridge, E. Barnes, C. L. Bethel, D. W. Carruth, M. Kocturova, M. Pleva, and J. Juhar, "Accessible electroencephalograms (eegs): A comparative review with openbci's ultracortex mark iv headset," *2019 29th International Conference Radioelektronika (RADIOELEKTRONIKA)*, pp. 1–6, 2019.

[262] B. Stern, *Inside the muse*, `https://learn.adafruit.com/muse-headset-teardown/inside-the-muse`, 2015.

[263] F. I. Database, *Emotiv epoc+ neuroheadset teardown internal photos*, https://fccid.io/2ADIH-EPOC02/Internal-Photos/Internal-Photos-2596557, 2019.

[264] A. Wipprecht, *Neurosky mindwave mobile teardown*, https://www.instructables.com/id/NeuroSky-MindWave-Mobile-Teardown-to-customized-EE/, 2014.

[265] G. Rosas-Cholula, J. Ramirez-Cortes, V. Alarcon-Aquino, P. Gomez-Gil, J. Rangel-Magdaleno, and C. Reyes-Garcia, "Gyroscope-driven mouse pointer with an emotiv® eeg headset and data analysis based on empirical mode decomposition," *Sensors*, vol. 13, no. 8, pp. 10 561–10 583, 2013.

[266] R. Mahajan and B. I. Morshed, "Unsupervised eye blink artifact denoising of eeg data with modified multiscale sample entropy, kurtosis, and wavelet-ica," *IEEE journal of Biomedical and Health Informatics*, vol. 19, no. 1, pp. 158–165, 2014.

[267] N. Naseer and K.-S. Hong, "Fnirs-based brain-computer interfaces: A review," *Frontiers in human neuroscience*, vol. 9, p. 3, 2015.

[268] J. Park, H. Kim, J.-W. Sohn, J.-r. Choi, and S.-P. Kim, "Eeg beta oscillations in the temporoparietal area related to the accuracy in estimating others' preference," *Frontiers in Human Neuroscience*, vol. 12, p. 43, 2018.

[269] T. Joachims, "Optimizing search engines using clickthrough data," *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 133–142, 2002.

[270] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the royal statistical society: series B (statistical methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[271] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," *Ijcai*, vol. 14, no. 2, pp. 1137–1145, 1995.

[272] W. B. Knox and P. Stone, "Reinforcement learning from simultaneous human and mdp reward.," *AAMAS*, pp. 475–482, 2012.

[273] R. S. Sutton, A. G. Barto, *et al.*, *Introduction to reinforcement learning*. MIT press Cambridge, 1998.

[274] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[275] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT Press, 2016.

[276] Y. Kassahun, B. Yu, A. T. Tibebu, D. Stoyanov, S. Giannarou, J. H. Metzen, and E. Vander Poorten, "Surgical robotics beyond enhanced dexterity instrumentation: A

survey of machine learning techniques and their role in intelligent and autonomous surgical actions," *International journal of computer assisted radiology and surgery*, vol. 11, no. 4, pp. 553–568, 2016.

[277] S. Banerjee, V. Dhiman, B. Griffin, and J. J. Corso, *Do deep reinforcement learning algorithms really learn to navigate?* 2018.

[278] A. P. Braga and A. F. Araújo, "A topological reinforcement learning agent for navigation," *Neural Computing & Applications*, vol. 12, no. 3-4, pp. 220–236, 2003.

[279] B. Bischoff, D. Nguyen-Tuong, I. Lee, F. Streichert, A. Knoll, *et al.*, "Hierarchical reinforcement learning for robot navigation," in *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2013)*, 2013.

[280] G. tech robotarium lab., "Http://www.news.gatech.edu/features/robotarium-robotics-lab-accessible-all.,"

[281] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," *arXiv preprint arXiv:1710.02298*, 2017.

[282] A. Irpan, *Deep reinforcement learning doesn't work yet*, `https://www.alexirpan.com/2018/02/14/rl-hard.html`, 2018.

[283] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, and P. Abbeel, "Reverse curriculum generation for reinforcement learning," *arXiv preprint arXiv:1707.05300*, 2017.

[284] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu, "Reinforcement learning with unsupervised auxiliary tasks," *arXiv preprint arXiv:1611.05397*, 2016.

[285] M. Gimelfarb, S. Sanner, and C.-G. Lee, "Reinforcement learning with multiple experts: A bayesian model combination approach," *Advances in Neural Information Processing Systems*, pp. 9528–9538, 2018.

[286] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.

[287] A. Y. Ng, S. J. Russell, *et al.*, "Algorithms for inverse reinforcement learning.," *Icml*, vol. 1, p. 2, 2000.

[288] W. B. Knox and P. Stone, "Augmenting reinforcement learning with human feedback," *ICML 2011 Workshop on New Developments in Imitation Learning (July 2011)*, vol. 855, p. 3, 2011.

[289] M. K. Scheffers, M. G. Coles, P. Bernstein, W. J. Gehring, and E. Donchin, "Event-related brain potentials and error-related processing: An analysis of incorrect re-

sponses to go and no-go stimuli," *Psychophysiology*, vol. 33, no. 1, pp. 42–53, 1996.

[290] G. Hajcak and D. Foti, "Errors are aversive: Defensive motivation and the error-related negativity," *Psychological science*, vol. 19, no. 2, pp. 103–108, 2008.

[291] W. H. Miltner, U. Lemke, T. Weiss, C. Holroyd, M. K. Scheffers, and M. G. Coles, "Implementation of error-processing in the human anterior cingulate cortex: A source analysis of the magnetic equivalent of the error-related negativity," *Biological psychology*, vol. 64, no. 1-2, pp. 157–166, 2003.

[292] G. Hajcak, J. S. Moser, N. Yeung, and R. F. Simons, "On the ern and the significance of errors," *Psychophysiology*, vol. 42, no. 2, pp. 151–160, 2005.

[293] K.-R. Müller, M. Tangermann, G. Dornhege, M. Krauledat, G. Curio, and B. Blankertz, "Machine learning for real-time single-trial eeg-analysis: From brain–computer interfacing to mental state monitoring," *Journal of neuroscience methods*, vol. 167, no. 1, pp. 82–90, 2008.

[294] A. Buttfield, P. W. Ferrez, and J. R. Millan, "Towards a robust bci: Error potentials and online learning," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 14, no. 2, pp. 164–168, 2006.

[295] R. Chavarriaga, P. W. Ferrez, and J. d. R. Millán, "To err is human: Learning from error potentials in brain-computer interfaces," pp. 777–782, 2008.

[296] M. G. Coles, M. K. Scheffers, and C. B. Holroyd, "Why is there an ern/ne on correct trials? response representations, stimulus-related components, and the theory of error-processing," *Biological psychology*, vol. 56, no. 3, pp. 173–189, 2001.

[297] S. Nieuwenhuis, K. R. Ridderinkhof, J. Blom, G. P. Band, and A. Kok, "Error-related brain potentials are differentially related to awareness of response errors: Evidence from an antisaccade task," *Psychophysiology*, vol. 38, no. 5, pp. 752–760, 2001.

[298] W. Gehring, M. Coles, D. Meyer, and E. Donchin, "A brain potential manifestation of error-related processing [supplement]," *Electroencephalography and clinical neurophysiology. Supplement*, vol. 44, pp. 261–72, Feb. 1995.

[299] M. Falkenstein, J. Hohnsbein, J. Hoormann, and L. Blanke, "Effects of crossmodal divided attention on late erp components. ii. error processing in choice reaction tasks.," *Electroencephalography and clinical neurophysiology*, vol. 78 6, pp. 447–55, 1991.

[300] M. Falkenstein, J. Hoormann, S. Christ, and J. Hohnsbein, "Erp components on reaction errors and their functional significance: A tutorial," *Biological Psychology*, vol. 51, pp. 87–107, Feb. 2000.

[301] W. H. R. Miltner, C. H. Braun, and M. G. H. Coles, "Event-related brain potentials following incorrect feedback in a time-estimation task: Evidence for a "generic" neural system for error detection," *J. Cognitive Neuroscience*, vol. 9, no. 6, 788–798, Nov. 1997.

[302] L. Osterhout, M. D. Allen, J. Mclaughlin, and K. Inoue, "Brain potentials elicited by prose-embedded linguistic anomalies," *Memory & Cognition*, vol. 30, no. 8, pp. 1304–1312, Dec. 2002.

[303] M. J. Maguire, G. Magnon, D. A. Ogiela, R. Egbert, and L. Sides, "The N300 ERP component reveals developmental changes in object and action identification," *Developmental Cognitive Neuroscience*, vol. 5, pp. 1–9, 2013.

[304] H. S. Squires NK Squires KC, "Two varieties of long-latency positive waves evoked by unpredictable auditory stimuli in man," *Electroencephalogr Clin Neurophysiol*, 1975.

[305] J. R. Folstein and C. Van Petten, "Influence of cognitive control and mismatch on the n2 component of the erp: A review," *Psychophysiology*, 2008.

[306] C. B. Holroyd and M. G. H. Coles, "The neural basis of human error processing: Reinforcement learning, dopamine, and the error-related negativity.," *Psychological review*, vol. 109 4, pp. 679–709, 2002.

[307] H. Niki and M. Watanabe, "Prefrontal and cingulate activity during timing behavior in the monkey," *Brain research*, vol. 171, pp. 213–24, Sep. 1979.

[308] H Gemba, K Sasaki, and V. Brooks, "
error' potentials in limbic cortex (anterior cingulate area 24) of monkeys during motor learning," *Neuroscience letters*, vol. 70, pp. 223–7, Nov. 1986.

[309] A. Barachant and M. Congedo, "A plug&play p300 bci using information geometry," *arXiv preprint arXiv:1409.0107*, 2014.

[310] B. Rivet, A. Souloumiac, V. Attina, and G. Gibert, "Xdawn algorithm to enhance evoked potentials: Application to brain–computer interface," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 8, pp. 2035–2043, 2009.

[311] A. Barachant and S. Bonnet, "Channel selection procedure using riemannian distance for bci applications," *2011 5th International IEEE/EMBS Conference on Neural Engineering*, pp. 348–351, 2011.

[312] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Classification of covariance matrices using a riemannian-based kernel for bci applications," *Neurocomputing*, vol. 112, pp. 172–178, 2013.

[313] C.-P. Lee and C.-J. Lin, "A study on l2-loss (squared hinge-loss) multiclass svm," *Neural computation*, vol. 25, no. 5, pp. 1302–1323, 2013.

[314] L. Bottou, "Large-scale machine learning with stochastic gradient descent," *Proceedings of COMPSTAT'2010*, pp. 177–186, 2010.

[315] B. Zadrozny and C. Elkan, "Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers," *Icml*, vol. 1, pp. 609–616, 2001.

[316] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 694–699, 2002.

[317] D. B. Percival, A. T. Walden, *et al.*, *Spectral analysis for physical applications*. cambridge university press, 1993.

[318] J. Platt *et al.*, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in large margin classifiers*, vol. 10, no. 3, pp. 61–74, 1999.

[319] D. Slepian, "Prolate spheroidal wave functions, fourier analysis, and uncertainty—v: The discrete case," *Bell System Technical Journal*, vol. 57, no. 5, pp. 1371–1430, 1978.

[320] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," *ICML*, vol. 99, pp. 278–287, 1999.

[321] K. Azizzadenesheli, E. Brunskill, and A. Anandkumar, "Efficient exploration through bayesian deep q-networks," *2018 Information Theory and Applications Workshop (ITA)*, pp. 1–9, 2018.

[322] B. D. Ziebart, "Modeling purposeful adaptive behavior with the principle of maximum causal entropy," Ph.D. dissertation, figshare, 2010.

[323] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement learning with deep energy-based policies," *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1352–1361, 2017.

[324] D. J. Krusienski, E. W. Sellers, D. J. McFarland, T. M. Vaughan, and J. R. Wolpaw, "Toward enhanced p300 speller performance," *Journal of neuroscience methods*, vol. 167, no. 1, pp. 15–21, 2008.

[325] J. Frey, "Comparison of an open-hardware electroencephalography amplifier with medical grade device in brain-computer interface applications," *arXiv preprint arXiv:1606.02438*, 2016.

[326] H. Takahashi, T. Yoshikawa, and T. Furuhashi, "Reliability-based automatic repeat request with error potential-based error correction for improving p300 speller performance," *International Conference on Neural Information Processing*, pp. 50–57, 2010.

[327] A. Kreilinger, C. Neuper, G. Pfurtscheller, and G. R. Müller-Putz, "Implementation of error detection into the graz-brain-computer interface, the interaction error potential," *European Conference for the Advancement of Assistive Technology*, pp. 195–199, 2009.

[328] M. J. Herrmann, J. Römmler, A.-C. Ehlis, A. Heidrich, and A. J. Fallgatter, "Source localization (loreta) of the error-related-negativity (ern/ne) and positivity (pe)," *Cognitive brain research*, vol. 20, no. 2, pp. 294–299, 2004.

[329] M. Agarwal and R. Sivakumar, "Charge for a whole day: Extending battery life for bci wearables using a lightweight wake-up command," *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2020.

[330] M. Agarwal and R. Sivakumar, "Cerebro: A wearable solution to detect and track user preferences using brainwaves," *The 5th ACM Workshop on Wearable Systems and Applications*, pp. 47–52, 2019.

[331] D. Xu, M. Agarwal, F. Fekri, and R. Sivakumar, "Playing games with implicit human feedback," *Workshop on Reinforcement Learning in Games, AAAI*, 2020.

[332] D. Xu, M. Agarwal, F. Fekri, and R. Sivakumar, "Accelerating reinforcement learning agent with eeg-based implicit human feedback," *arXiv preprint arXiv:2006.16498*, 2020.

[333] M. Agarwal, S. K. Venkateswaran, and R. Sivakumar, "Human-in-the-loop rl with an eeg wearable headset: On effective use of brainwaves to accelerate learning," *Proceedings of the 6th ACM Workshop on Wearable Systems and Applications*, pp. 25–30, 2020.