Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Jostein Nordal Lysberg

Deep Learning for Cognitive Load Classification on a Multimodal Eye-Tracking Dataset

Master's thesis in Department of Engineering Cybernetics
Supervisor: Sverre Hendseth
June 2022

NTNU
Norwegian University of
Science and Technology

Jostein Nordal Lysberg

Deep Learning for Cognitive Load Classification on a Multimodal Eye-Tracking Dataset

Master's thesis in Department of Engineering Cybernetics
Supervisor: Sverre Hendseth
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU

Kunnskap for en bedre verden

Preface

This thesis is submitted as part of the master's degree requirements at the Department of Engineering Cybernetics at the Norwegian University of Science and Technology. The work presented has been carried out under the supervision of Assoc Prof. Sverre Hendseth at the Department of Engineering Cybernetics, NTNU.

The thesis is, in some fashion, a continuation of a specialization project produced during the autumn of 2021. This project was not published, so any background theory originating from it will be restated in full throughout the text to maintain the reading experience. Specifically, this includes the following sections:

- Chapter 1: Sections 1.2.1 and 1.2.2.
- Chapter 2: Sections 2.3 and 2.5.

During the project, the author was fortunate enough to get in touch with Prof. Mila Vulchanova and Prof. Giosuè Baggio from the Department of Language and Literature at Dragvoll. They manage the Language Acquisition and Language Processing Lab, which could provide a Tobii Pro Spectrum for eye-tracking data collection. Additionally, the Tobii Pro Lab software was used to design a recording environment.

Finally, the Pytorch Deep Learning Library [1] was used to streamline all machine learning development, training, evaluation, and testing.

Unless otherwise stated, all figures and illustrations have been created by the author.

Acknowledgements

First, I would like to thank my best friend and companion, Sophie Stokker, for being supportive and encouraging, despite my frequent silent treatments after a disappointing day of writing. Next, my fellow graduates deserve thanks for creating a work environment where writing a thesis has felt like a breeze. Finally, a special thanks go to Mila and Giosuè for taking me seriously and for showing genuine interest in what I was exploring. This project would never reach its conclusion if not for all the help, insight, and facilities provided.

Jostein N. Lysberg
Trondheim, June 2022

Abstract

This project explores deep learning for cognitive load classification using a multimodal eye-tracking dataset. The work presented is motivated by an emerging market for data-driven performance analytics and health surveillance in gaming.

Electronic Sports (esports) is a growing industry, and as kids and adults get progressively into competitive gaming, the need for concrete feedback for performance advancements arises. However, most good feedback comes from direct coaching, which is no reasonable alternative for the casual gamer. With the emergence of commercially available eye-tracking, the solution presents itself as an automated performance distinction method, using ocular data to infer insights into cognition.

The solution is enabled by the ocular system's immense implication on cognition. Cognitive load, in particular, has shown impressive correlations with pupil size, Spontaneous Eyeblink Rate (EBR), and gaze patterns. A set of state-of-the-art deep learning architectures was explored, developed, and compared for Time Series Classification (TSC). Eye-tracking data was recorded in an environment with a carefully controlled task to reliably capture cognitive load as ground truth in a training dataset. The dataset was subsequently labeled by task state and difficulty level and used to train classification models.

The dataset displayed correlations remarkably consistent with the literature. Although limited in quantity and generalizability, it exhibited features distinct enough to train neural networks for intra-task and intra-subject classification. In the end, the best models could distinguish between four states of task exposure with an accuracy of 71% and three levels of cognitive load with an accuracy of 61%. These results serve as proof of concept. They lay the foundation for further research on non-intrusive means for performance analytics. Future work should address dataset creation and generalizability to generate models that can reliably distinguish cognitive load between subjects and on any task.

Sammendrag

Dette prosjektet utforsker dyp læring for klassifisering av kognitiv belastning ved å bruke et multimodalt øyesporingsdatasett. Arbeidet som presenteres er motivert av et fremvoksende marked for datadrevet ytelsesanalyse og helseovervåking innen spill.

Elektronisk Sport (e-sport) er en voksende bransje, og etter hvert som flere barn og voksne begynner å konkurrere, oppstår behovet for konkrete tilbakemeldinger for å fremme prestasjon. De fleste gode tilbakemeldingene kommer imidlertid fra direkte coaching, som ikke er noe fornuftig alternativ for hvem som helst. Med fremveksten av kommersielt tilgjengelig øyefølging, presenterer løsningen seg selv som en automatisert ytelsesforskjellsmetode, som bruker okulære data for å utlede innsikt i kognisjon.

Løsningen er muliggjort av det okulære systemets enorme tilknytning til kognisjon. Spesielt kognitiv belastning har vist imponerende korrelasjoner med pupillestørrelse, spontan øyeblinkfrekvens og blikkmønstre. Et sett med toppmoderne modeller for dyp læring ble utforsket, utviklet og sammenlignet for klassifisering av tidsserier. Øyefølgingsdata ble tatt opp i et miljø med en nøye kontrollert oppgave for å pålitelig fange opp kognitiv belastning som målklasser i et treningsdatasett. Datasettet ble deretter merket etter oppgavetilstand og vanskelighetsgrad og brukt til å trene klassifikasjonsmodellene.

Datasettet viste korrelasjoner som var bemerkelsesverdig i samsvar med litteraturen. Selv om den var begrenset i mengde og generaliserbarhet, viste den funksjoner som var distinkte nok til å trene nevrale nettverk for innad-oppgave og innad-bruker klassifisering. Til slutt kunne de beste modellene skille mellom fire tilstander av oppgaveeksponering med en nøyaktighet på 71% og tre nivåer av kognitiv belastning med en nøyaktighet på 61%. Disse resultatene beviser at konseptet fungerer. De legger grunnlaget for videre forskning på ikke-påtrengende midler for prestasjonsanalyse. Fremtidig arbeid bør ta for seg datasettdesign og generaliserbarhet for å generere modeller som pålitelig kan skille kognitiv belastning mellom brukere og på enhver oppgave.

Contents

Acronyms

BL Binary Level. 35, 52, 57

BS Binary State. 35, 51, 57, 58

CLT Cognitive Load Theory. ix, 3, 5, 6, 29

CNN Convolutional Neural Network. ix, 24, 25, 40, 57, 59

DA Dopamine. 11, 14, 15

EBR Spontaneous Eyeblink Rate. v, ix, x, 14, 15, 17, 28, 30, 33, 35, 37, 38, 54, 55, 56, 59, 60

EEG Electroencephalogram. 6, 28, 29, 31

esports Electronic Sports. v, ix, 2

FCN Fully Convolutional Network. 40, 41, 43, 49, 50, 51, 52, 57, 70

FL Full Level. 35, 52, 57

FS Full State. 35, 49, 50, 51, 57

LC Locus Coeruleus. 11, 12

MCDCNN Multi-Channel Deep Convolutional Neural Network. 41, 49, 57, 71

MLP Multi-Layer Perceptron. 23, 25, 26, 39, 40, 41, 49, 57, 69

NASA TLX NASA Task Load Index. 6, 29

NE Noradrenaline. 11, 12, 15

PLR Pupil Light Response. 9, 10, 11, 29, 31, 54

PNR Pupil Near Response. 9, 10, 11, 29, 31, 54

PNS Parasympathetic Nervous System. 11, 12, 54

PPR Psychosensory Pupil Response. 9, 10, 11, 12

ReLU Rectified Linear Unit. 23, 39, 40, 41, 43

ResNet11 11-Layer Residual Network. 42, 43, 49, 50, 51, 57, 72

ResNet18 18-Layer Residual Network. 42, 43, 49, 52, 57, 73

SNS Sympathetic Nervous System. 11

TSC Time Series Classification. v, 4, 38, 39, 43, 57, 59

Chapter 1

Introduction

1.1 Collaboration

This project is a research collaboration with Osirion AS. Osirion is a newly founded team with a passion for gaming and data-driven performance analytics. We want to leverage eye-tracking and other physiological measures to facilitate sustainable and healthy gaming routines for average and professional consumers. Users should be able to effortlessly improve their skills while monitoring their eye health, sleep, mental fatigue, and more. The work presented will contribute to developing a platform where all of this is possible.

1.2 Motivation

1.2.1 Eye-Tracking as an Emerging Technology

Eye-tracking hardware has been available for a long time. Studies dating as far back as the 80s show the use of eye trackers together with computers, which achieved accuracies up to half a degree of visual angle [2]. Even before cameras and computers were advanced enough to measure anything accurately, mirrors have been used in reading exercises to observe gaze patterns and cognitive behavior [3]. The current state-of-the-art eye trackers provide massive accuracy improvements, higher sampling frequencies, better data quality, stability, and ease of use.

Most eye trackers in wide adoption today are used by researchers for research purposes, with data quality and price tags fit only for large research budgets. As an effect, manufacturers have had little incentive to promote the commercial availability of the hardware and its accompanying software. However, recent trends in the market have enabled the emergence of much more affordable eye-tracking. This trend promotes its use even for the casual user, which has remodeled the commercial approach to eye-tracking applications.

Traditionally, the leading value proposition for eye-tracking has been as an assistive technology for people with disabilities, offering an improvement to the

autonomy and quality of life for those in need of alternate input devices [4, 5]. More recently, the video-game industry has caught wind of the technology through a series of very promising studies over the past decades [6–8]. These studies suggest that eye-tracking hardware need not directly substitute existing control input devices but could instead serve to complement them. For example, game developers can make game graphics more immersive by letting the user's gaze point determine camera focus, depth of field, or light exposure. Game characters may interact differently with the user depending on whether they maintain eye contact. Eye-tracking provides a more challenging and immersive experience [9], and its adoption is only going to increase as the technology and its applications advance in the future.

1.2.2 Eye-Tracking in Electronic Sports

The ever-expanding competitive gaming environment drives another compelling use case for eye-tracking. As the video game industry is already worth more than the music and movie industries combined [10], all estimates show a positive trend in the interest for Electronic Sports (esports). In fact, market reports show that the esports audience reached 474 million people million in 2018, with a year-on-year growth of +8,7% [11]. With this impressive growth comes the ever-increasing demand for competitive performance analytics.

There is always an incentive to be better at whatever game one plays, especially if the competitive scene is attractive. Naturally, the most effective method of improving performance is through direct feedback. Many amateur and pro players tend to subscribe to software services that provide targeted match feedback, as is evident by the success of companies such as Mobalytics and Shadow Esports. At Osirion AS, we aim to complement existing applications of match feedback with that which can be inferred from the analysis of eye-tracking data.

1.2.3 Cognitive Load

To reliably give targeted feedback to the user, we first need ways of distinguishing the good players from the great. Only then can we begin considering the aspects that separate them and help novice players reach higher levels of performance.

As the French poet Guillaume de Salluste so eloquently portrays them, our eyes can be considered "windows of the soul" [12] for their broad implications on cognition. As such, when eye-tracking is available as a data source, cognitive load is a natural step towards performance distinction. Tamara Van Gog, professor of educational sciences at the department of education at Utrecht University, states the following. "Eye tracking is not only a useful tool to study cognitive processes and cognitive load in computer-based learning environments but can also be used indirectly or directly in the design of components of such environments to enhance cognitive processes and foster learning." [3]. In a report, she refers to several studies where eye-tracking implementations have increased successful problem-solving.

It is safe to assume that a given task becomes decreasingly demanding with continued practice and increased experience. World-famous psychologist and Nobel Prize winner Daniel Kahneman [13] published a best-selling pop-science book in 2013, where he depicts two systems that drive the way we think. According to Kahneman [13], the fast-thinking "System 1" is the most efficient actor when complex tasks are to be executed, as it is guided by intuition. This intuition serves to ease task execution such that capacity is freed from the slow-thinking conscious self. The catch, however, is that intuition needs to be trained. The field of psychology, as mentioned above, is commonly known as Cognitive Load Theory (CLT) and will be explained in detail in section 2.1. In short, cognitive load is a vastly complex metric that is subject to many confounding variables. Therefore, it is challenging to measure directly, and the development of accurate methods is an open problem.

As will become apparent in section 2.2, ocular measures made available by modern eye-tracking have clear correlations with cognition. If a classification model could accurately predict levels of cognitive load from eye-tracking data, there is great potential for further research in user-targeted feedback in esports. Moreover, combining cognitive load with performance metrics may allow for the calculation of an index of cognitive capacity, mental efficiency, task expertise, or even intellect [14].

1.3 Research Goals and Objectives

While the end goal for Osirion is to provide a platform that facilitates performance and health in gaming, this thesis is merely a means toward that end. To reliably produce feedback causally linked with skill, we need a systematic understanding of all aspects that may ultimately affect gameplay. Such aspects are inevitably both environmental and subject to circumstance. However, as the above section suggests, the significance of the cognitive load aspect should not be underestimated. Since emerging technologies allow for increasingly unobtrusive and commercially available eye-tracking, bridging the gap between such data and cognitive load classification may prove a significant value for performance analytics. These grounds raise the following research goals:

- Give a comprehensive literature review of cognitive load theory and subsequent correlations with ocular events
- Design and train a machine learning model to classify cognitive load from features in eye-tracking data

Any model that can perform reliable classification with accuracies beyond chance will be considered a success. An exploration of model architectures will be provided and compared to reasonably exclude any doubt of individual weaknesses in model properties. The author has no access to pre-made datasets on which these models may be trained. Therefore, a part of this project will regard the considerations and manual development of such a dataset.

The first goal is included to substantiate any results given by the second goal by taking a theoretical approach to the same problem. The author will consult the literature to provide the tools with which a compelling conclusion may be made. It will also allow for more justified considerations in model architecture, dataset, and data recording environments.

There are many steps along the way towards these goals. A rough indication of the objectives along the way is detailed below.

1. Consult literature in neuroscience and cognitive psychology to fundamentally understand cognitive load and its implications.
2. Make an informed decision on the creation of a training dataset. That includes answering the following. How can cognitive load be induced in the subject and labeled? Which data channels should be included? What considerations need to be taken with regards to the recording environment?
3. Design an experimental setup and record eye movement data.
4. Process raw data and create a training dataset.
5. Consult literature in data science and recent advances in Time Series Classification (TSC) and choose a set of model architectures to implement.
6. Implement and train model architectures.
7. Consider classification results and their implications and value for future work in automated performance analytics for gaming.

Chapter 2

Background

2.1 Cognitive Load Theory

Cognitive Load Theory (CLT) as a field of study was first developed by Sweller [15] as a means of optimizing intellectual performance in learning environments. By categorizing differing levels of strain imposed on the user in various problem-solving tasks, he could reason between the effectiveness of different information presentation formats for learning. He elaborates his theories a decade later in a paper outlining a system for human cognitive architecture. With this system, he provided tools to describe the nature of mental processing demands, intellectual skill, and the ability to learn.

2.1.1 What is Cognitive Load?

Cognitive load is a broad term aimed at describing the load imposed on the cognitive system while performing a particular task [16]. Considering the complex nature of the cognitive system and all factors both affecting and affected by cognitive load, the term is necessarily very comprehensive. To fully understand the nature of cognitive load, we need to describe it by its defining factors.

One such factor is the distinction between *mental load*, *mental effort* and *performance*. Although similar on the surface, these three dimensions cause much difficulty when assessing cognitive load.

Consider the case of one participant in an experimental trial where researchers want to measure cognitive load. The participant is presented with one or more tasks, each with varying difficulty levels. The task is then said to impose a mental load, and the amount of cognitive resources the participant genuinely allocates to perform this task is defined as mental effort [17]. Whereas mental effort is dynamically determined by the cognitive control and focus of the participant, the mental load will be constant with every task. Furthermore, when researchers want to measure the cognitive load imposed, a performance metric is often employed. Although there is a clear relationship between high cognitive load and low performance, performance measures may, in practice, show small deficits with increas-

ing task demands because the participant is able to invest more mental effort to compensate for increasing mental load [18].

Consequently, the intensity of effort expended (mental effort) is often considered the essence of cognitive load [19], since this metric may yield important information that is not necessarily reflected in performance-based measures. The measurement of mental effort is no trivial task and has long been dependent on subjective self-assessment methods such as NASA Task Load Index (NASA TLX) [20]. Given that changes in cognitive functioning are often reflected in physiology, later years have seen the emergence of more objective physiological measures. Some are based on Electroencephalogram (EEG), eye-tracking, and galvanic skin response, to name a few. However, since their correlation with cognition is often complex, expert domain experience or machine learning is required to interpret the data. The potential for some physiological data sources is detailed in section 2.2.

2.1.2 Three Domains of Cognitive Load

Human cognitive architecture can be defined by working memory capacity and the relative load imposed by incoming information. Sweller [15] proposed a system similar to figure 2.1a to describe the typical flow of information in the human information processing system. As is apparent from the figure, there is a constant interplay of information to and from the sensory-, working-, and long-term memory. Both sensory input and long-term encoded information are processed in working memory. The ease with which information processing in working memory occurs is a prime concern of CLT and is often condensed into three domains.

First, the information presented can be intrinsically hard to process, often due to a high level of element interactivity. The more elements that need to be kept in memory at one time, the higher the mental load. For example, learning a new language by memorizing individual words has a low level of element interactivity since only a pair of words need to be kept in memory to learn their association. In contrast, learning a language also requires grammatical knowledge. In this case, relationships in entire sentences require multiple words and meanings to be kept in memory at one time, representing a high level of element interactivity. The cognitive load imposed by the underlying nature of the information is called *intrinsic cognitive load*.

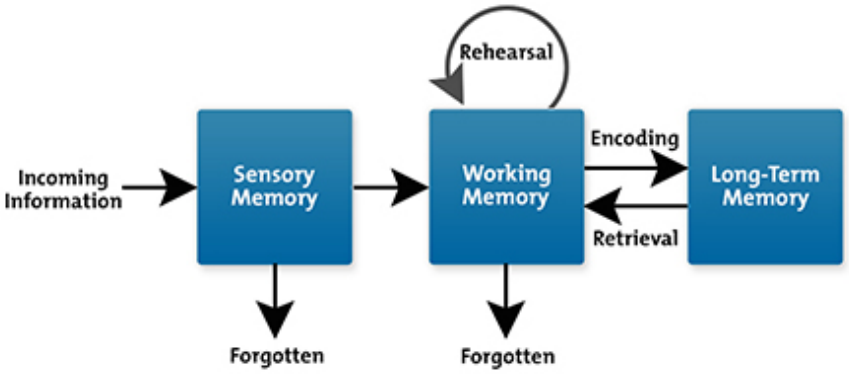
On top of the intrinsic cognitive load of information is its *extraneous cognitive load*. This domain depends on how information is presented, whether intuitively or unintuitively. Consider for example the presentations of figure 2.1a compared to figure 2.1b. They represent the same information and hence impose the same intrinsic cognitive load. Yet, the second figure places higher demands on working memory by forcing the viewer to memorize labels and is, therefore, more difficult to process. Extraneous cognitive load is a fundamental concern when designing learning environments where one can choose between presentation formats. One generally wants the extraneous cognitive load to be as low as possible in order

to optimize learning and make room in working memory for what is known as *germane cognitive load*.

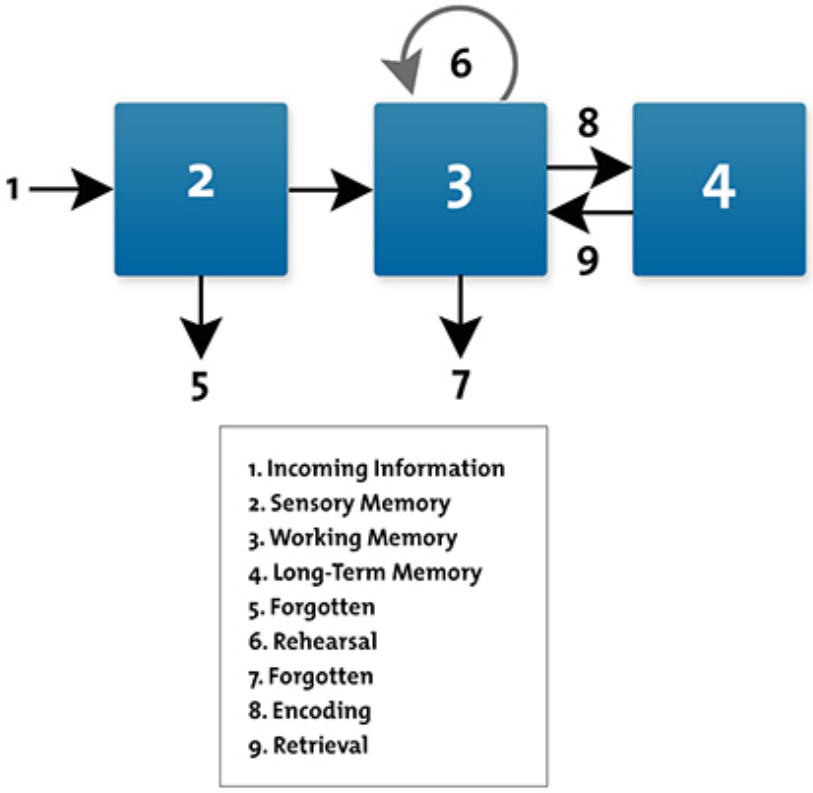
Within human cognitive architecture, Sweller [15] establishes that understanding is the process of encoding complex information and relationships so that they can effectively be stored in long-term memory. Information is compressed into frameworks of understanding called *schemas*. This understanding is decoded from memory for reuse when such information is later encountered anew. For example, when fresh mathematicians are to learn to multiply out the denominator in an equation such as $a/b = c$, they have to hold several concepts in working memory. For one, they need to know that b can be multiplied onto both sides of the equation without affecting a. Second, the concept that $b/b = 1$ must be readily known, and third that $a * 1 = a$. For someone with no prior knowledge of algebra, this is a problem with high element interactivity and likely demands a high degree of mental effort to solve. However, a learned student can merely shift b to the right side of the equation, which gives cb. Learning and understanding the procedure substantially reduces the cognitive load imposed by the task.

Germane cognitive load is the load imposed on working memory from constructing these schemas. When designing learning environments, Sweller [15] therefore recommends that "the learner's attention must be withdrawn from processes not relevant to learning and directed toward processes that are relevant to learning and, in particular, toward the construction and mindful abstraction of schemas." For example, a student is more prone to understand a complex topic through instructional procedures such as asking questions about a problem rather than memorizing their solutions directly. With active engagement in learning, conscious effort in working memory will be made to create topic understanding.

In summary, master chess players easily outmatch multiple novice players in simultaneous games without breaking a sweat. A professor considers a complex topic trivial, while students scratch their heads in confusion. These examples do not mean that the expert chess player or the professor necessarily has a much higher cognitive capacity or intelligence than the novice or the student, but rather that they possess a heightened understanding. Understanding is information compressed to schemas in long-term memory, which allows topics to demand much less intrinsic cognitive load when recalled. Experience is acquired through years of conscious effort, likely with a high degree of germane cognitive load imposed on working memory.



(a) Intuitive presentation of the information processing system.



(b) Less intuitive presentation of the information processing system.

Figure 2.1: Example of how two presentations of the same information impose different levels of extraeneous cognitive load.

Source: [21]

2.2 Cognitive Impacts on the Ocular System

As the reader might agree, vision may be the most vital sense of perception that humans enjoy. We can interpret vast amounts of information every second from the raw signals of millions of optic nerve fibers. This process is so complex that a significant fraction of the cortex is involved [22]. As such, it is natural to believe that our eyes are a good metric when studying the internal cognitive functions of our brain. In fact, the use of eye movements to learn from the inner workings of the mind has been exploited by cognitive psychologists for over two centuries [23].

This section will outline the most prominent impacts of cognition on the ocular system, where there are clear traces of correlation between the mind and the eyes. Some such correlations have clear causal links to neural circuitry and chemical release in the brainstem, while others can be determined by empirical evidence from clinical- and pharmacological studies.

2.2.1 Pupillometry

Voluntary effort drives visual perception. Except for reflexive responses, eye movements require conscious activation of the extraocular muscles surrounding the eyes. These are necessary movements for placing visual information on the retina and are well-understood. Even after attention is directed at an object, other eye movements constantly operate to provide our brain with an optimal image of reality. One such operation includes lens stretching to provide focused light from near and far objects. Another is responsible for light admission to the retina by constricting and dilating the pupil. From a neurological examinator's standpoint, the latter is of particular interest. Pupillometry is the study of pupil size and reactivity and will be further elaborated on in the current subsection.

Our brain regulates pupil diameter as a response to three distinct types of stimuli. Brightness and near fixations constrict its size, and cognitive activity dilates it. These types of stimuli will be referred to as the Pupil Light Response (PLR), Pupil Near Response (PNR) and Psychosensory Pupil Response (PPR), respectively. Although not particularly interesting in the larger context of this thesis, the effects of PLR and PNR must still be understood when assessing the cause of a given pupillary response.

Pupil Light Response (PLR) and Pupil Near Response (PNR)

Average pupil size ranges from 1.5mm in bright lighting to 9mm in total darkness [24]. The physiological explanation for this effect is clear: A large pupil allows for more light to fall on the retina, thus increasing the amount of information sent to the brain at any time. Photoreceptors on the retina quickly become saturated upon bright light exposure, rendering them less sensitive. Subsequent exposure to faint objects in dim light requires dark adaptation, taking tens of minutes. During this time, vision is substantially impaired. To avoid this issue, the pupil may rapidly

dilate upon changing environments from bright to dim, effectively acting as a buffer for light admittance. This behavior is subconsciously controlled by the PLR.

Diverting gaze between near and far objects causes an additional few millimeters of changes in pupil size. This response is explained by optical distortion of light passing through the lens, caused by *spherical aberration*. This distortion occurs because the light entering an aperture is increasingly refracted toward the edges of a convex lens, causing rays not to converge on the same focal point. The effect is illustrated in figure 2.2. Like with a conventional camera, less spherical aberration means that deeper planes of the visual field may be in focus simultaneously. Reducing the amount of light that has to enter through the lens edges by reducing the aperture size, therefore, deepens the field of focus.

Furthermore, since focusing on closer objects require a rounder lens causing even more distortion, optimal vision at close range benefits from a constricted pupil. As with the PLR, this behavior is subconsciously controlled by the PNR.

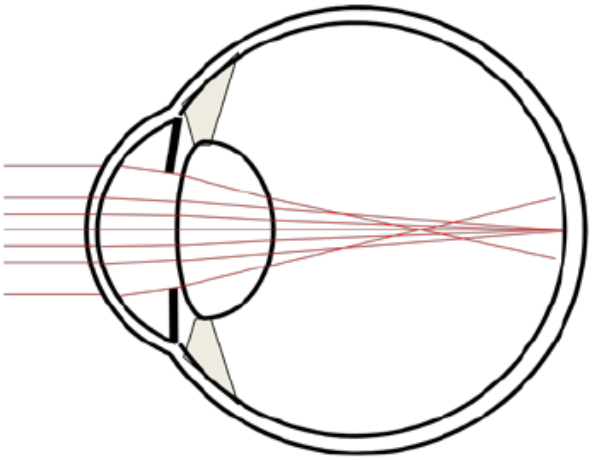


Figure 2.2: Simplified illustration of spherical aberration in the eye. Light passing through the outer edges of the lens are refracted to a larger degree than those passing through the center, causing image distortion.

Source: [25]

Whereas both the PLR and the PNR can cause pupil size to vary by half or double its original size, PPR only accounts for changes of less than 0.5mm [26]. Naturally, this leads to significant sources of bias if the cognitively evoked responses are what we are interested in measuring. Researchers may mitigate the effects of PNR by keeping the stimulus at a constant distance from the subject, which is usually the case for most studies that present stimuli on a computer monitor. To account for the PLR, studies suggest moderately and constantly lit environments [27]. By closely controlling these two factors and recording a baseline pupil diameter for each subject, researchers may observe the PPR by measuring deviations from this baseline.

Psychosensory Pupil Response (PPR)

While the PLR and the PNR offer rather clear-cut causes for their effect (e.g. light intensity or distance to fixation), the PPR is not that simple. One cannot link pupil size to just one cognitive process. A multitude of mental mechanisms, such as alertness, arousal, mental effort, and cognitive capacity, have all been linked to pupil dilation in some way [26, 28, 29]. Eckstein *et al.* [24] states that "pupil dilation reflects a specific, intensity- and attention-related aspect of cognitive processing." In other words, any cognitive process that requires heightened attention or deliberate effort will affect pupil size.

Two distinct central- and autonomous nervous system pathways govern pupil dilation and constriction. One is responsible for the fight-or-flight response and is referred to as the Sympathetic Nervous System (SNS). The other is called the Parasympathetic Nervous System (PNS) and controls what is known as the rest-and-digest response [30]. As their responses might suggest, they operate under entirely different circumstances. When chemicals in the brainstem stimulate the SNS, heart rate, sweat- and glucose production is accelerated, and pupils dilate. Conversely, stimulation of the PNS will constrict the pupil, slow the heartbeat, and trigger digestion, secretion, and voiding. This connection hints at why pupils are small at rest and larger when aroused or agitated.

Furthermore, there is a cluster of neurons in our brain called the Locus Coeruleus (LC). These neurons have shown direct inhibitory projections with the PNS and excitatory projections with the SNS [24]. In other words, increased activity in the LC triggers bodily functions associated with fight-or-flight while halting those associated with rest-and-digest. These responses further lead to reduced activation of the pupil's constricting fibers and increased activation of its dilating fibers, thus increasing its size. A study by Rajkowski [31] demonstrated this effect, where they recorded neuronal LC activity in monkeys during a target detection task. During this task, monkeys were made to exert different levels of effort over time. As is evident from figure 2.3, a striking temporal coupling between LC activity and pupil size was shown.

To explain why the PPR has any coupling with cognition, besides being associated with the fight-or-flight response, we need to understand the Locus Coeruleus. Studies show that the LC plays a critical role in physiological arousal and cognitive functioning. In fact, the LC is the only source of Noradrenaline (NE) in the cerebral cortex [32]. NE, synthesized from Dopamine (DA), is an essential neuromodulator of brain activity [33]. Although it produces many effects in the body, the most notable is boosting the signal-to-noise ratio of incoming sensory information.

This effect has been demonstrated in spatial working memory tasks in monkeys, where chemicals that inhibited NE release was administered to observe their effect on task performance. The task given was to search through a number of boxes and find target objects. In subsequent trials, the monkeys were rewarded for remembering previous targets. The neural circuitry of working memory associated with the target responses had increased activation with higher levels of NE.

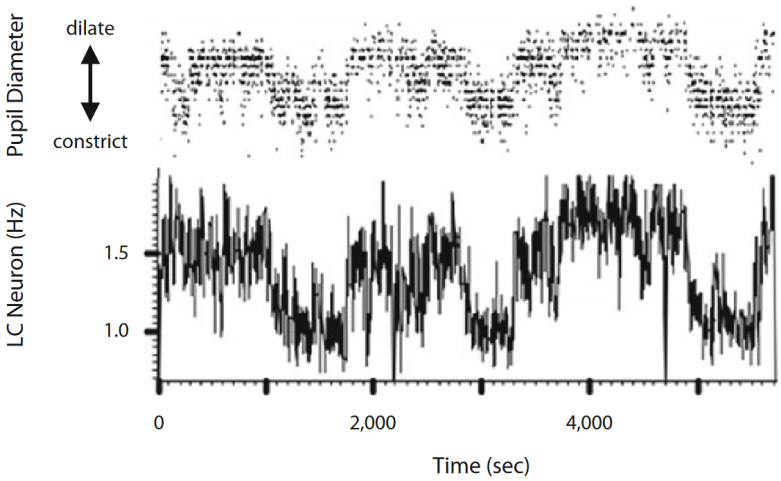


Figure 2.3: Locus Coeruleus firing rate (bottom) and pupil diameter in monkeys (top). Demonstrates clear connection between LC and PNS inhibition, which we know regulates pupil diameter.

Source: [31]

Conversely, activation of the neural circuitry surrounding this particular response was reduced [34]. In other words, the monkey was more likely to select a response corresponding to the target when NE release was higher. Higher firing rates of the LC therefore made "signals" from the neural circuitry corresponding to the target response strengthened compared to the "noise" from incorrect responses.

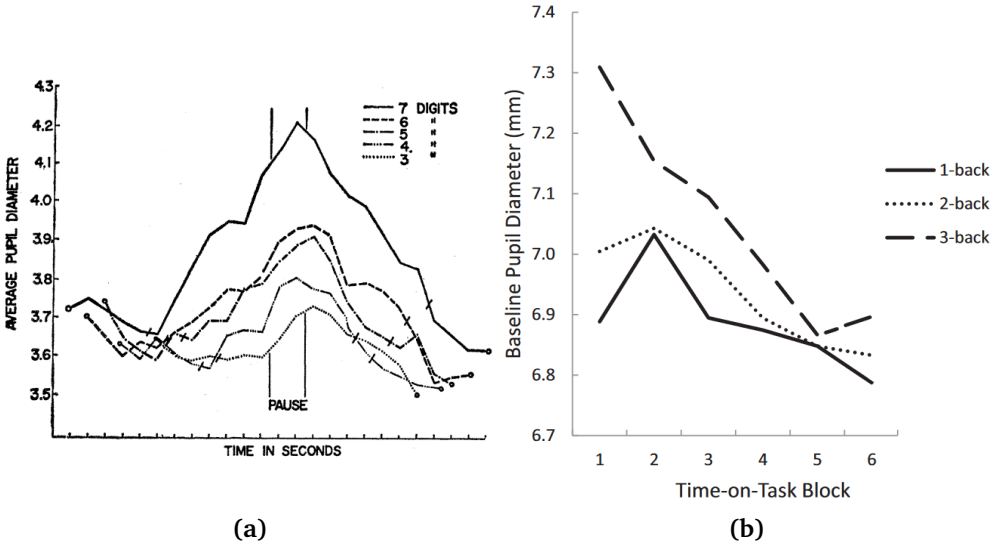
In summary, research shows that we can use NE quantities in the brainstem as a measure of how narrow or broad the attentional focus is. Attentional focus may encapsulate a variety of complex cognitive functions, such as mental workload, working memory capacity, decision making, and much more [32]. Since NE is directly mediated by the Locus Coeruleus (LC), which has neural links with segments of the central nervous system governing pupil size, PPR may be used as a proxy for such cognitive functions.

Experiments

A famous experiment by Kahneman and Beatty [35] was one of the first which demonstrated pupillometry as a measure of cognition. In this study, pupil diameter was shown to reflect the amount of material that was under active processing at any time. They employed several short-term memory tasks, one of which is reproduced in figure 2.4a. For this particular task, strings of varying lengths of digits were presented to the subject. After a two-second pause, they were instructed to repeat them from memory. Naturally, longer strings imposed a higher load on working memory than shorter ones. This relation shows a significant correlation with average pupil diameter. Interestingly, the pupil seems to dilate for every new digit presented, peaking just before recall begins. The pupil then constricts for

every digit recalled, returning to baseline after the task is completed.

In addition to this task-evoked response, another experiment by Hopstaken *et al.* [36] also demonstrate that cognitive load markedly affects the pupil diameter baseline. This study employed the N-back task, a task which will be explained in much more detail in section 3.2. In short, it is designed to be simple in principle and very cognitively demanding in practice. It requires the sustained engagement of working memory and attention at varying difficulty levels from simple (0-back) to very hard (3-back). As is apparent from figure 2.4b, baseline pupil diameter was consistently larger with increased task difficulty. This experiment also demonstrates how fatigue from continued task exposure affects task engagement. Each time-on-task block on the x-axis represents 18 minutes of task exposure. As is evident from the decrease in baseline pupil diameter for each block, task exposure also has a noticeable effect on pupil size. This relation could, in theory, be leveraged as an indicator of task engagement or mental fatigue.



(a) **(b)**

Figure 2.4: a) Average pupil diameter during a short-term memory task. Each line represent one length of string for recall. Tick marks before and after the pause indicate the beginning of string presentation and end of recall, respectively. Points are calculated from an average over all subjects at one time instance. **b)** Average baseline pupil diameter during three difficulty levels of the n-back task. Each graph represent one value of N (i.e., task difficulty level). Points are calculated from an average of all subjects and all time instances during 6 minutes of task exposure for each difficulty level.

Source: [35, 36]

2.2.2 Spontaneous Eyeblink Rate (EBR)

Another well-understood ocular event is blinking. It serves eye health by distributing an even layer of moisture across the eyeball and protects against foreign objects by reflexive closure. Contrary to the pupillary response, blinking can be controlled by conscious effort. Therefore, when considering the cognitive implications of blink rate, we need to define *spontaneous eyeblinks*. What characterizes these eyeblinks is that cognitive functions subconsciously induce them without interference from volition. They differ from reflexive eyeblinks because they occur at predictable rates and without the triggers of foreign objects.

Spontaneous Eyeblink Rate (EBR), the rate at which spontaneous eyeblinks occur, is a reliable measure of Dopamine (DA) activity in the central nervous system. Although the precise neural circuitry that controls blink rate is primarily unknown [24], empirical studies have repeatedly demonstrated this relationship.

DA activity in both human and non-human primates can reliably be observed by either pharmacological manipulations or from clinical studies. For instance, administration of the well-known DA agonists *apomorphine* and *amphetamine* have shown statistically significant increases in EBR in both humans [37–39], and monkeys [40, 41]. Apomorphine effect on EBR is shown in figure 2.5. Similarly, many conditions are known to affect the dopaminergic system. One such condition is Parkinson's disease, a disorder caused by a loss of dopaminergic cells in some parts of the brain. Patients suffering from Parkinson's show a significantly reduced EBR compared to healthy individuals [42]. Other conditions, such as schizophrenia and Tourette's syndrome, are linked with elevated DA activity and show increased EBR.

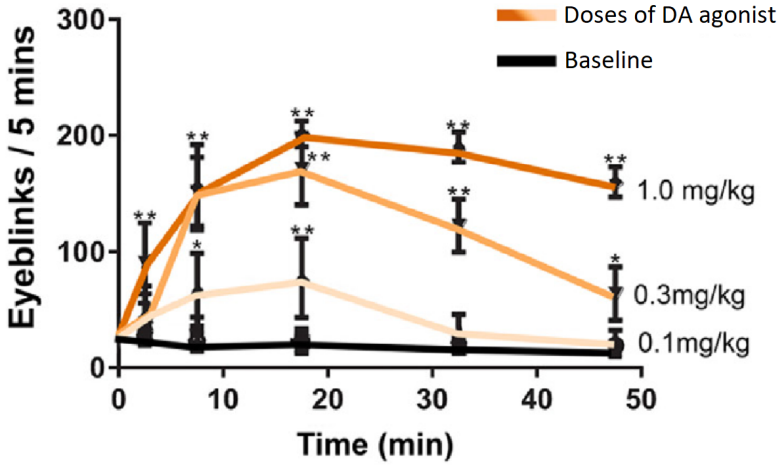


Figure 2.5: Spontaneous Eyeblink Rate for three dosages of DA agonist apomorphine against baseline (no pharmacological manipulation). Note that EBR values on y-axis show deviations from baseline, hence why baseline graph is constantly at 0.

Source: [41]

Just like NE, DA is an important neuromodulator for brain activity. Especially for learning and *cognitive control*, DA activity has shown to be an important aggregator. Cognitive control is a term that broadly describes our ability to control impulses, maintain goals and focus on specific tasks without diverting attention. Research in this area has shown a near positive-linear relationship between baseline DA levels in the brainstem and cognitive control in goal-oriented tasks [43, 44]. In addition to baseline levels, phasic DA activity has shown positive correlations with changing task environments. Bochove *et al.* [45], for instance, showed that an increased EBR in one task trial could reliably predict the level of cognitive control and performance in the subsequent trial.

Experiments

A classic experiment to ascertain cognitive control is called the Stroop test. It is set up such that the subject is presented with a series of words representing colors (e.g., red, green, blue). Each word stimulus is said to be *congruent* if the font color matches the word. If colors do not match, the stimulus is *incongruent*. Incongruent stimuli are notably challenging since the subject is presented with two conflicting visual inputs, and an intuitive answer does not come to mind without some thought. Numerous studies, including the original by Stroop [46], show that the incongruent condition causes an increased response time of about 75% compared to the congruent condition. More recent studies suggest that conflicts in the word read and the color seen requires higher levels of cognitive control to respond correctly [47, 48].

An experiment by Oh *et al.* [49] employed the Stroop test to examine task-evoked EBR, similar to Bochove *et al.* [45]. In this study, subjects were presented with 60 randomly distributed congruent and incongruent stimuli. Each stimulus stayed on-screen for two seconds or until the subject responded with a color. They found that, of the 27 subjects in the experiment, each subject could be divided into distinct subgroups based on their eyeblink behavior when responding. Seventeen showed a consistent tendency to blink directly before responding (subgroup I), and seven directly after (subgroup II). Four subjects showed both behaviors (subgroup III). As can be seen from figure 2.6, eyeblinks were much more likely to occur in the ±500ms surrounding the subject response. Subgroups I and II demonstrated this effect even more clearly, and together, they represent an 88% majority. For all three subgroups, this suggests that spontaneous eyeblinks are closely associated with the exertion of cognitive control and may signal a shift in the subjects' internal cognitive or attentional state.

During the same study by Oh *et al.* [49], baseline EBR levels were measured during a two-minute resting period both between trials and on-task. As can be seen from figure 2.7, the color-naming task produced a slightly higher EBR than the word-reading task. Furthermore, the average on-task EBR turned out to be almost 50% elevated from the resting baseline. These results suggest that EBR may not only indicate shifts in cognitive control but also sustained attention.

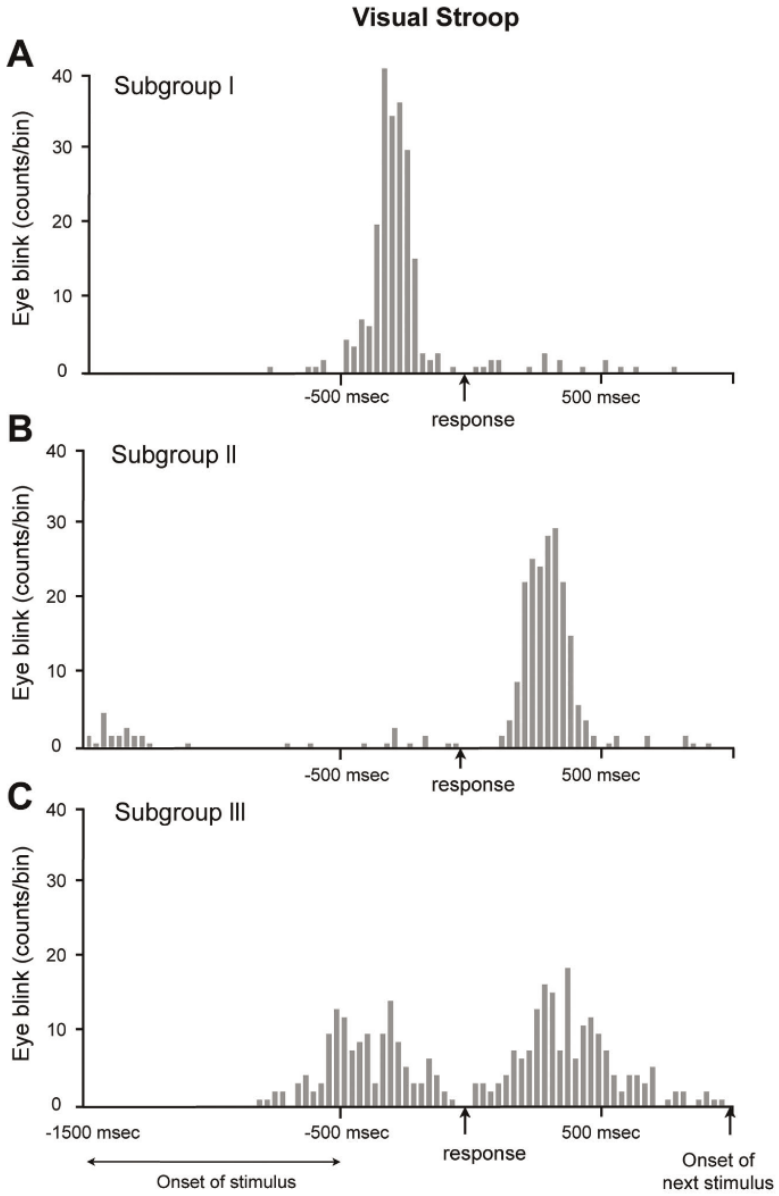


Figure 2.6: Histogram showing spontaneous eyeblink count as a response to the Stroop test. Each bin (point on the x-axis) represents 30-millisecond intervals. An arrow indicates the time of response. Due to randomness, stimulus onset is between -1500ms and -500ms from the response. Onset of the next stimulus occurs after a one seconds waiting period.

Source: [49]

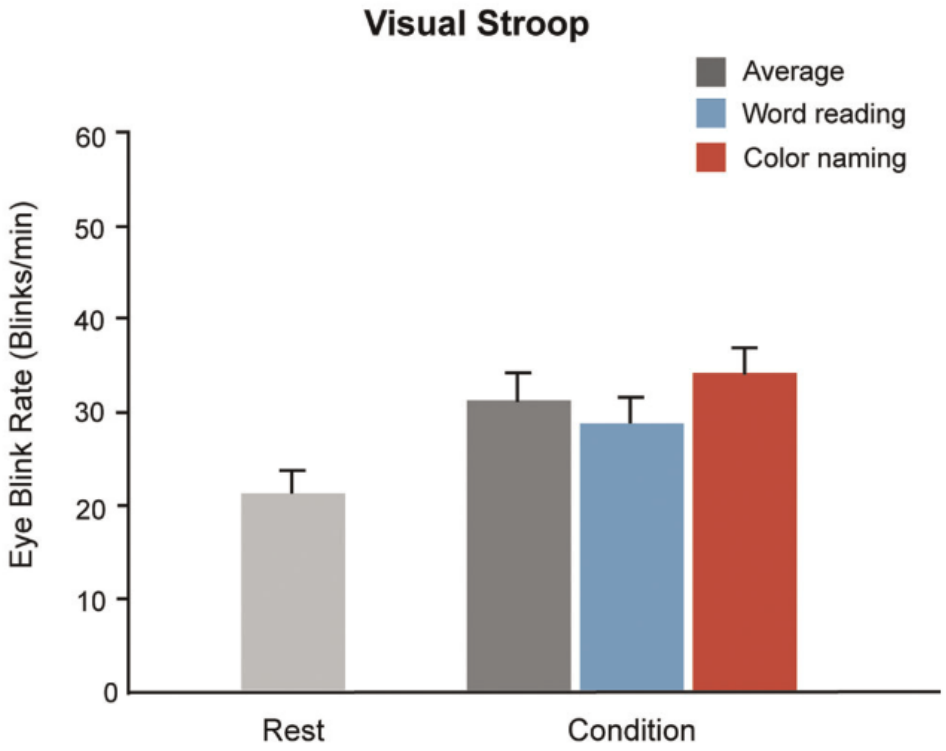


Figure 2.7: Spontaneous Eyeblink Rate (EBR) during the Stroop test. The resting eyeblink rate in light grey was recorded during a 2 minute long idle period between trials. Word reading and word naming conditions represent trials where the target response was the word of the stimulus and the color of the stimulus, respectively.

Source: [49]

2.2.3 Eye Gaze

Eye gaze comprises all voluntary and involuntary eye movements besides those mentioned above. Fixations, saccades, and microsaccades are examples, but so are all features that they induce: Saccadic frequency, fixation duration, field-of-view, and so on. Although voluntary in nature, many of these elements are in reality governed by subconscious cognitive aspects. It is these correlations we will attempt to uncover in this subsection. As for the spontaneous eyeblink rate, the neural circuitry underlying these events is difficult to pinpoint. There are, however, empirical studies that show promising correlations.

Experiments

A series of interesting studies by Williams [50–52] suggest that task-induced cognitive load may adversely affect the subject's field of view, as measured by information perceived in the visual periphery. Such results are hard to measure accurately, as gaze patterns are immensely complex and may introduce significant confounding variables. He employed a task where subjects were required to report the orientation of lines presented at varying distances from a fixation point. At central vision, they were required to perform a short-term memory task with varying levels of induced cognitive load. Williams [51] found that the accuracy of determining line orientation deteriorated rapidly with increasing cognitive load in the primary task. He argues that this suggests a human tendency to subdue our field of view when highly concentrated or under a heavy mental workload.

Another study by Underwood *et al.* [53] demonstrated that varying levels of cognitive demand during a driving task affected the individual's search patterns. They managed this by instructing a group of subjects to drive a car through a set route, during which their eye movements would be recorded for three separate one-minute intervals. The intervals were designed to represent three levels of demand based on the type of road they were on. One was a one-lane rural road with good visibility. Another was a busy suburban road with much traffic and pedestrians. The third was a high-speed dual-carriageway joined by two slip roads from the left and right.

As can be seen from the results presented in figure 2.8, the types of roads significantly affect fixation durations and both vertical and horizontal search patterns. It seems that fixation durations decrease on the more demanding roads while the search strategy widens, indicating a cognitive effect on gaze. These results are reasonable, as shorter duration and more frequent fixations are naturally associated with heightened awareness. The wider search patterns are also natural since demanding roads require attention to the information in a larger field of view.

The same study by Underwood *et al.* [53] also researches the effect of experience on search patterns. It is interesting to note that while experienced drivers expand their field of view when encountering the more demanding roads, novice drivers tend to maintain one level of scanning throughout. This result relates to the theory derived in section 2.1.2, which connects experience with the construction of mental schemas. Such schemas allow for fewer demands on working memory, which is subsequently apparent in gaze patterns.

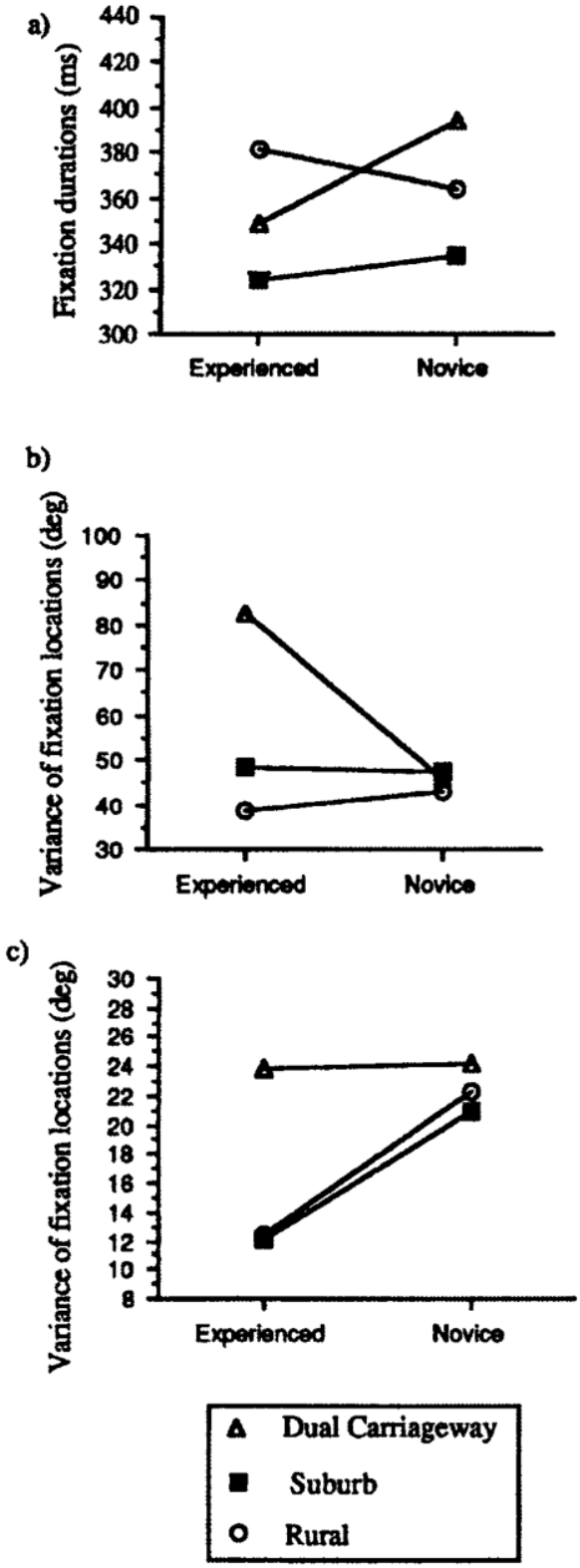


Figure 2.8: Gaze patterns observed on subjects in a driving task. Each plot is labeled by the type of road driven on. Each road type infers differing levels of demand from the driver. **a)** Mean fixation durations. **b)** Horizontal variance in fixation locations **c)** Vertical variance in fixation locations.

Source: [53]

2.3 Machine Learning Fundamentals

Artificial intelligence is a field of research that has been practiced for a very long time, contrary to popular belief. The first work that is now recognized as AI was written by McCulloch and Pitts [54]. Even still, the concept of artifacts operating under their own control can be traced back to Alexandria ca. 250 BC, where a water regulator was built that could maintain a constant flow rate. Other examples of self-regulating feedback control systems include the steam engine governor and the thermostat, all invented before the 19th-century [55]. This thesis will concentrate on machine learning, a particular form of artificial intelligence that employs prior knowledge of historical data to tackle tasks that are too difficult to solve with fixed programs written and designed by human beings [56].

2.3.1 Learning Algorithms

A machine learning algorithm is characterized by the fact that it can learn properties from data. Mitchell [57] famously defines this aspect of learning as "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." Since machine learning algorithms can be employed in a wide range of problems and trained by an equally wide range of methods, T, P, and E in this definition can be constructed as just about anything. However, a very common class of tasks T central to this thesis is that of *classification*. The experience E required to train towards this task is usually provided through either *supervised-* or *unsupervised learning*.

Classification

Classification is the task of determining which of k discrete categories some input belongs. Given data (experience E) produced by a function $f : \mathbb{R}^2 \to \{1, \ldots, k\}$, a learning algorithm tasked with classification will generate a hypothesis $h : \mathbb{R}^2 \to \{1, \ldots, k\}$ that approximates f. Given an input vector $\boldsymbol{x}$, $h(\boldsymbol{x})$ outputs a probability distribution over possible categories. One popular application is that of object detection, where an image is given as the input $\boldsymbol{x}$, and the output is the category to which an object in the image belongs. If $k = 2$, the learning problem is called binary classification, and the hypothesis h merely outputs the probability of whether an input represents a single target category or not. Multi-class classification is more common in the general case, however.

Since we often cannot assume any prior knowledge of the inherent properties of f, choosing a hypothesis h with which to approximate it is no trivial task. We say that h is selected from a *hypothesis space* $\mathcal{H}$, which needs to be defined by the learning algorithm designer. Looking at figure 2.9, one can see the importance of choosing a hypothesis space that is complex enough to approximate f accurately yet simple enough that the function does not overfit. For this example, a 12-degree polynomial is required to produce an approximation that perfectly agrees with all

the data. However, since we cannot assume that f is not stochastic, this polynomial might generalize poorly to unseen data. In such a case, a simpler hypothesis in a linear or a sinusoidal function might be the optimal choice.

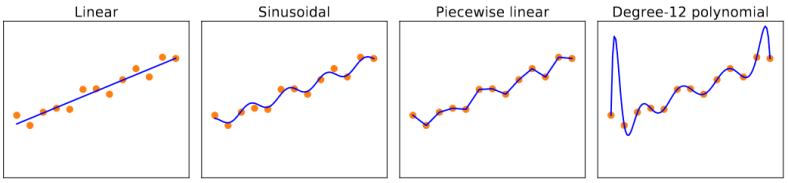


Figure 2.9: Finding hypotheses to fit data. The four plots each show best-fit functions from four different hypothesis spaces trained on the same dataset.

Source: [55]

Supervised Learning

A machine learning algorithm can be called a supervised learning algorithm if it is trained by experiencing a dataset of examples $\boldsymbol{x}$, where each example is associated with a target y. The training process thus becomes to approximate a function that can reproduce y given $\boldsymbol{x}$.

One classic learning algorithm is linear regression, which is supervised learning in its simplest form [56]. The goal of linear regression is to predict a target value $\hat{y} \in \mathbb{R}$ from an n-dimensional input vector $\boldsymbol{x} \in \mathbb{R}^n$. Since we know then that the output should be a linear function of the input, the problem becomes to approximate the hypothesis given by equation 2.1. Here $\boldsymbol{w} \in \mathbb{R}^n$ is a vector of model parameters, which control the behavior of the system.

$$h(\boldsymbol{x}) = \hat{y} = \boldsymbol{w}^T \boldsymbol{x} \tag{2.1}$$

Now, to determine the optimal value of $\boldsymbol{w}$, we need a performance measure P. For this particular problem, a typical choice is the *mean squared error* (MSE), given in equation 2.2. MSE is calculated from a matrix of m input vectors $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$, their corresponding m target values $\boldsymbol{y} = \{y_1, \ldots, y_m\}$, and model parameters $\boldsymbol{w}$.

$$MSE = \frac{1}{m} \sum_i (\hat{\boldsymbol{y}} - \boldsymbol{y})^2 = \frac{1}{m} ||\hat{\boldsymbol{y}} - \boldsymbol{y}||_2^2 \tag{2.2}$$

By setting $\hat{\boldsymbol{y}} = \boldsymbol{y}$, one can see that $MSE = 0$, and furthermore that it increases linearly with the euclidean distance between $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$. As such, the optimal model parameters can be obtained by minimizing MSE with respect to $\boldsymbol{w}$. This can be done by solving for where its gradient is $\boldsymbol{0}$, as is in equations 2.3 to 2.6.

$$\nabla_{\boldsymbol{w}} MSE = \boldsymbol{0} \tag{2.3}$$

$$\Rightarrow \nabla_{\boldsymbol{w}} \frac{1}{m} ||\hat{\boldsymbol{y}} - \boldsymbol{y}||_2^2 = \boldsymbol{0} \tag{2.4}$$

$$\Rightarrow \nabla_w \frac{1}{m} ||\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}||_2^2 = \boldsymbol{0} \tag{2.5}$$

$$\vdots$$

$$\Rightarrow \boldsymbol{w} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{2.6}$$

Evaluation of equation 2.6 results in a value of $\boldsymbol{w}$ which optimally fits the training dataset. As such, it constitutes a simple learning algorithm [56].

2.4 Deep Learning

Deep learning is a powerful extension of the simple learning algorithms detailed above. In theory, any problem which can be modeled as the mapping of an input vector to an output vector can be solved by deep learning [56]. In other words, any function of any complexity can be approximated given sufficiently large models and sufficiently large labeled training examples.

2.4.1 Deep Feedforward Neural Networks

Feedforward neural networks constitute a branch of deep learning covering models with a one-directional data flow from input to output. They are called *neural networks* because they are loosely inspired by the neural circuitry of the human brain. Extending this analogy, all human perception may be considered one vast deep feedforward neural network. It works by mapping sensory input from sight, touch, smell, taste, and hearing through countless layers of neurons to output what we perceive of the world. Artificial neural networks work similarly by mapping a vector of input data points to one or more output values.

Even in their simplest form, deep feedforward neural networks overcome many challenges that linear models face. Linear regression, for instance, may never accurately approximate a non-linear function or capture interactions between two or more input variables. This limitation may only be overcome by introducing a nonlinearity to the input $\boldsymbol{x}$. We encapsulate this nonlinearity as $\phi(\boldsymbol{x})$. Thus, the equivalent to equation 2.1 becomes equation 2.7.

$$y = f(\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{x}) = \phi(\boldsymbol{x}; \boldsymbol{\theta})^T \boldsymbol{w} \tag{2.7}$$

We now have another set of trainable parameters $\boldsymbol{\theta}$, meaning that the final form of ϕ is yet to be determined. By allowing the model to experience a given set of labeled training data, $\boldsymbol{\theta}$ will gradually converge towards a value that enables the full model to approximate the data optimally. The parameters defined by $\boldsymbol{w}$ may subsequently map from $\phi(\boldsymbol{x})$ to the desired output. ϕ is called a *hidden layer* because its outputs are unrelated to either the input or output vectors. It can instead be thought of as a transformation of the input layer. Its output is hence a new representation or interpretation of the information provided by $\boldsymbol{x}$.

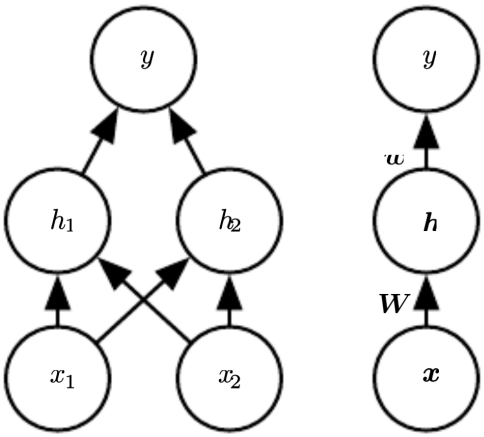


Figure 2.10: Example of a simple MLP. Left representation depicts individual neurons of each layer and the edges between them. The right representation is more common, depicting each layer with vector notation.

Source: [56]

Multi-Layer Perceptron

A Multi-Layer Perceptron (MLP) is one form of feedforward neural network, the simplest implementation of which is illustrated in figure 2.10. Here we have two input neurons, one hidden layer with two neurons, and one output layer with one neuron. With vector notation, we say that the hidden layer $\boldsymbol{h}$ is a function of the input layer $\boldsymbol{x}$, with the relation $\boldsymbol{h} = f^{(1)}(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{c})$. $\boldsymbol{h}$ is then the input to a third layer (which in this case is also the output layer) with the relation $y = f^{(2)}(\boldsymbol{h}; \boldsymbol{W}, \boldsymbol{c})$. We can keep adding depth and width to the model this way, further increasing its complexity and potential hypothesis space. The entire model may now be given by equation 2.8.

$$f(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{c}, \boldsymbol{w}, b) = f^{(2)}(f^{(1)}(\boldsymbol{x})) \tag{2.8}$$

To provide the nonlinearity we want, we define $\boldsymbol{h} = g(\boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{c})$. This functon is commonly known as the *activation function* of the layer. For deeper networks, it would be added to the output of every hidden layer. There are many activation functions to choose from, but the default recommendation in the machine learning community is to use the Rectified Linear Unit (ReLU) [58]. It is defined as $g(z) = max\{0, z\}$ and has been made popular for its simplicity without trading off on optimization with gradient-based backpropagation. The complete network of figure 2.10 can thus be given by equation 2.10.

$$y = f(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{c}, \boldsymbol{w}, b) = \boldsymbol{w}^T g(\boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{c}) + b \tag{2.9}$$

$$y = f(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{c}, \boldsymbol{w}, b) = \boldsymbol{w}^T max\{\boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{c}\} + b \tag{2.10}$$

2.4.2 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a specialized kind of deep feedforward neural network that has shown remarkable results in computer vision and language processing. When first proposed by LeCun [59] in 1989, it revolutionized machine learning problems where data had a grid-structured topology. Such data could be interpreted with more insight while allowing for much deeper and scaleable model architectures. Before its inception, most practitioners had little faith in the advent of neural networks for machine learning. The success of CNNs triggered a wave of interest, which laid the foundation for further research in neural networks and deep learning [56].

Convolution

To understand CNNs, we need to understand what a *convolution* is. Used in many engineering disciplines, it is simply a mathematical operator which expresses how two functions overlap [60]. In signal processing, for instance, convolution may be used to mathematically replicate how a given sound signal would behave in any conceivable environment. All one would need is the impulse response of this environment, which could be the recording of a clap or similar fast transient sound. Convolving these two signals would have the effect of "placing" the sound in this environment, artificially reproducing all reverberations. The operation is typically denoted with an asterisk and is defined as equations 2.11-2.12. This particular version is known as discrete convolution, which is what we will be encountering for neural networks.

$$s(t) = (x * k)(t) \tag{2.11}$$

$$s(t) = \sum_{a=-\infty}^{\infty} x(a)k(t-a) \tag{2.12}$$

With CNN terminology, x is the input to a convolutional layer and k is its *kernel*. Since the input to a CNN is often an image, time series, or other signals, x and k are usually multidimensional. Taking image classification as an example, the convolutional operations performed on all input pixels would look like equation 2.13.

$$S(i,j) = (K * X)(i,j) = \sum_{m}\sum_{n} X(i-m, j-n)K(m,n) \tag{2.13}$$

Each pixel is denoted by i and j for its spatial coordinate in the input image. The output from convolving a kernel K with an image X is called a *feature map* because it is essentially a mapping to a new "image" where features represented by the kernel are highlighted. A visual representation of how a 2-dimensional convolutional operation may be applied to an input is displayed in figure 2.11.

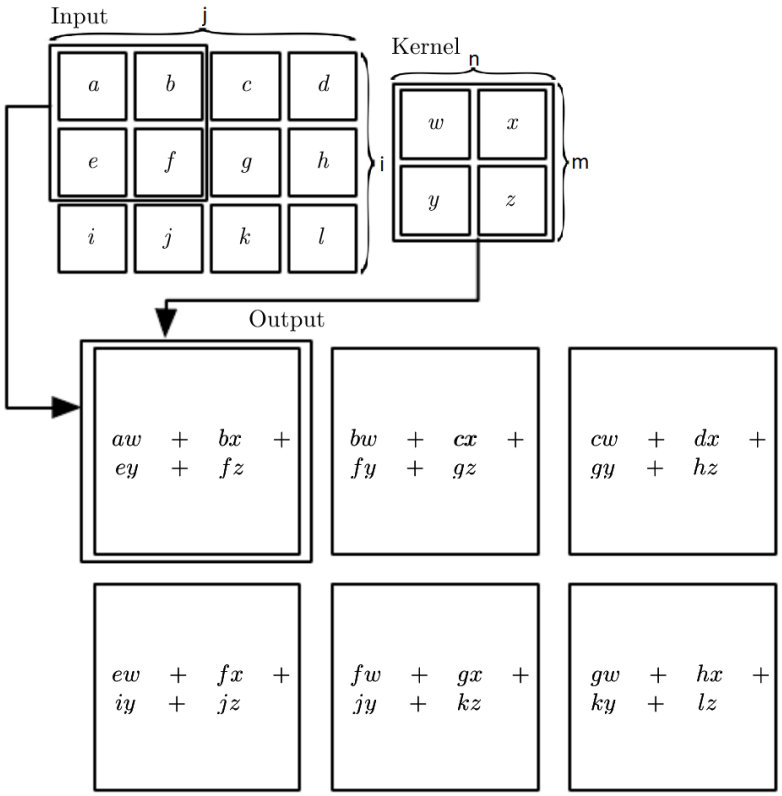


Figure 2.11: Example of 2-dimensional convolution operation. If this were image classification, values in the input matrix would represent individual pixel activations in the image. The final value of each output cell is calculated by the convolution operation given by equation 2.13.

Source: [56]

The kernel K is what defines all properties of a convolutional model. Instead of learning individual parameters for every edge between every neuron, as is the case for MLPs, CNNs learn the internal weights of all kernels in the network. As we will see, CNNs usually contain several kernels for every convolutional layer, each capturing one feature in the input to each layer.

Properties of CNNs

What makes CNNs so efficient is based on three essential ideas. First, since parameters are only present in each kernel, as described above, the network does not need a single parameter for all neurons in every layer. This idea is known as *parameter sharing* and will reduce the storage requirements of large models. Another consequence of the convolutional layers is that each input neuron is only connected to a few outputs. As can be seen in figure 2.11, only four neurons of the input are required to calculate the value of a cell in the output. This idea is called

sparse interactions. Contrary to an MLP, which requires connections from every input to every output, this feature drastically improves both computing efficiency and memory requirements during training.

Finally, a concept known as *equivariant representations* allows for the detection of features that are independent of spatial (or temporal) position in the input. In other words, if there is an object in the input to a convolutional layer that causes high activations in a feature map, the same object will cause an equally high activation in the same feature map if it were in another position. This effect is another consequence of the way in which feature maps are created by "moving" the same kernel across the entire input. MLPs, in contrast, lack this property since each neuron is associated with an individual learned weight.

2.5 Eye Tracking Technology

Eye-tracking systems have existed in some way since the late 1800s [61]. The first occurrences included bite-bars to ensure still head positions and mechanical rings attached to the eyeball. More modern techniques are based on electromagnetic inductance in a specially constructed lens, others directly on the electromagnetic activation of the extraocular muscles. The latter is more commonly known as electrooculography and is still present today in some systems. Ever since the appearance of the pupil- and corneal reflection method, the nature of which will be explained shortly, that has been the primary approach in all modern eye-tracking systems.

The dominance of the pupil- and corneal reflection method comes from its minimally intrusive manner, as it allows for precise and accurate gaze estimation from a video recording of the user's eye movements. The hardware of such eye-trackers consists of at least one high-resolution and high-frequency camera, accompanied by one or more infrared light sources directed at the user. These light sources will produce an infrared reflection on the cornea, which, together with the pupil, serves as reference points to determine gaze direction and head position. An example is illustrated in figure 2.12. Positions of reference points in the image are determined using computer vision algorithms. A calibration process is initially required to provide the system with known reference point relationships corresponding to known gaze points in the tracking area. The rest of the tracking area is then interpolated between calibration points. While even one camera and one light source provide a reasonably accurate gaze position as long as the head is fairly still, more cameras and infrared sources can be used to relax the constraints on head movement and calibration [61].

All eye-tracking methods suffer from a deficiency in degrading estimation with large gaze angles. In the extremes, the corneal reflection is often lost. Eye-trackers utilizing the pupil- and corneal reflection method rarely support gaze angles beyond 40° in the horizontal direction and 25° in the vertical direction. Given the viewing distance, d and the gaze angle θ, the corresponding unit x in the stimulus space is given by equation 2.14. Note, however, that this relationship only holds

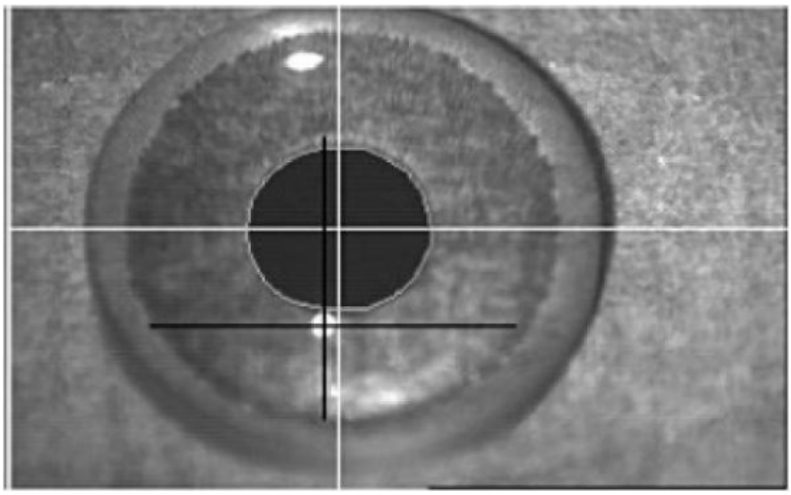


Figure 2.12: Demonstration of pupil- and corneal reflection. The white and black crosses correspond to the point indentified as the pupil and corneal reflection centers, respectively.

Source: Holmqvist *et al.* [61]

when θ is small, i.e., when the user looks at points close to the tracker camera. As such, the same visual angle θ_1 may result in different displacements (x_1 and x_2) on the stimulus for progressively larger gaze angles, as illustrated in figure 2.13.

$$\tan\frac{\theta}{2} = \frac{x}{d} \tag{2.14}$$

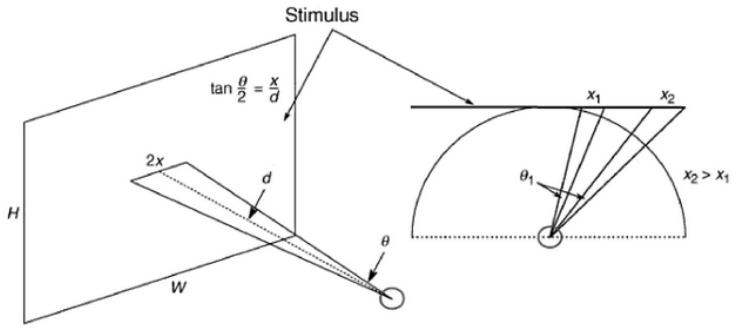


Figure 2.13: Geometric relationship between stimulus and degrees of visual angle.

Source: Holmqvist *et al.* [61]

All eye trackers come with a given sampling rate, and the speed at which data is recorded is typically the primary contributor to its price. Considering eye movements as an oscillating behavior, one can use the Nyquist-Shannon sampling theorem to argue that a sampling frequency at least twice as large as the largest

frequency component of the signal is required. It means that a sampling rate of 300Hz is necessary to estimate the movements of micro-saccades, which can oscillate at 150Hz. Still, for most use cases, it is often sufficient only to evaluate the more significant eye movements such as saccades, fixations, and smooth pursuits, which are not oscillating and subsequently require less strict sampling rates.

2.6 Related Work

Performance analytics in gaming is not the only motivation for cognitive load classification. Many related studies have done similar work for different purposes. Heard *et al.* [62] presents a review of 24 workload assessment algorithms, citing a use case for supervisory control environments such as the NASA control room for the Mars rover or a nuclear power plant. In such environments, one could imagine an adaptive workload system that dynamically monitors each employee's workload and changes autonomy or reassigns tasks accordingly. He argues that such systems could improve both workplace efficiency and employee health. Gerjets *et al.* [63] presents a similar adaptive system for instructional design in digital environments. Instead of improving workplace efficiency, such a system would improve learning efficiency. Real-time assessment of cognitive workload could be used to actively adapt the presentation of instructions such that learning is optimized by maximizing germane cognitive load.

While some alternative algorithms were presented in the review by Heard *et al.* [62], the predominant method of cognitive load classification has been with machine learning models. A study by Hogervorst *et al.* [64] compared the classification accuracies of various models on many combinations of data channels in a multimodal dataset. Data were recorded from the N-Back task, similar to the one described in section 3.2. They found that models trained on data from eye-tracking, EEG, and various physiological sensors achieved classification accuracies of over 90%. Furthermore, models trained on eye-tracking data alone (pupillary, EBR, and gaze) achieved accuracies up to 70%. Similar results were found by Lobo *et al.* [65] and Wilson *et al.* [66].

However, what limits cognitive load estimation in its current form is generalization across tasks and subjects. As concluded in the review by Heard *et al.* [62], training a model to distinguish cognitive load in more than one setting will severely limit its sensitivity since the impacts of cognition on physiology is complex and subjective. This is a tradeoff that must be assessed for every use case.

Generalization in cognitive load classification using eye-tracking was addressed in a dissertation by Appel *et al.* [67]. He made a comprehensive effort to understand which features of eye-tracking data were most likely to generalize well and developed some methods which improved generalization accuracy. A result of particular interest was that pupil diameter and fixation frequency responded similarly for most participants in the training dataset. This suggests that the choice of data channels is more critical if generalization is a priority.

Chapter 3

Implementation

3.1 Data Acquisition

Section 1.3 made the point that the end goal for this thesis is to classify cognitive load in humans from eye-tracking data. Unlike similar classification problems where data is readily available, a core challenge towards this goal lies in acquiring an eye-tracking dataset that reliably encapsulates differing levels of cognitive load. For reasons detailed in section 2.1, the nature of CLT makes this a particularly difficult problem.

The only measures of cognitive load which can truly be considered a "ground truth" in this regard are subjective self-assessment questionnaires [68, 69]. Such measures, like the NASA TLX, make accurate predictions of the cognitive load induced by a task through a long series of subjective questions and answers. Of course, the problem with such methods is that they are not viable for real-time estimation. There are less intrusive physiological measures available [70], such as EEG and heart rate. Still, even these can only ultimately be a proxy that is more or less on par with pupillometry alone.

As such, the most reasonable approach for the purposes of this thesis will be to manually record eye-tracking data in a setting where the task performed may be strictly controlled. Datasets can then be labeled by the difficulty and internal states of the task. In the light of considerations mentioned in section 2.2.1, a few assumptions need to be made for this approach to be viable:

1. Task difficulty levels reliably reflect its load on working memory.
2. The subject will always exert a mental effort to match the mental workload induced by the task.
3. Cognitive load may be evaluated by mental effort.
4. Mental effort remains fairly constant throughout task execution and between two or more sessions.
5. Data is not contaminated by environmental responses, such as the PNR and PLR.

Of these assumptions, assumption 4 may be the most uncertain. It is natural to believe that the subject's performance likely will increase with practice and decrease with fatigue. A given mental workload is known to induce reducing levels of mental effort with task experience [18], as detailed in section 2.1.2. Fatigue may also affect task engagement, reducing the motivation to exert maximum mental effort on every trial. Since both subjective experience and fatigue are hard to control, we can only accept these limitations for what they are and account for them when results are to be considered.

3.2 N-Back

3.2.1 Overview

Section 2.2 listed a number of standard experiments that are known to produce some form of mental strain on the subject. These experiments have been used extensively in many research fields, so their effects are constant and familiar. Therefore, they may be considered a sufficient proxy for cognitive load for this thesis. We will take inspiration from both Hopstaken *et al.* [36] and Appel *et al.* [67] and employ the N-Back task. It was chosen for its distinct levels of difficulty, which could be directly translated to levels of cognitive load for classification. Besides, its nature is simple, straightforward to implement, and requires nothing but a stimulus screen and an input device to deploy.

The N-Back task operates by displaying a series of stimuli in the form of single letters on the screen, switching letters every other second. With each presentation, the subject is asked to indicate whether the stimulus matches the target N screens prior by a keypress. Naturally, task difficulty increases with increasing N, demanding more and more information to be kept in working memory at any one time. The subject must constantly allocate attention to the task to perform well. Multiple studies have shown that both pupil diameter and EBR increase with increasing N [36, 71–73], proving its relevance for cognitive load classification.

3.2.2 Details

For this particular use case, the author chose three difficulty levels; N=0, N=1, and N=2. Many related experiments also employ N=3. However, experience shows that most participants tend to find this level too difficult and give up early [74, 75]. Additionally, since this thesis merely aims to classify cognitive load, three levels would be more than sufficient.

The first level (N=0) is intended to impose no particular cognitive load on the subject besides requiring sustained attention. It is achieved by showing the target stimulus before the task commences. This way, working memory only needs to memorize one constant target. For levels N=1 and N=2, the target stimuli are dynamically determined by the stimulus N screens prior.

Experimental sessions are structured so that the subject is exposed to continuous engagement for only five minutes at a time. Every session consists of three N-Back blocks, each split into three block segments for every task difficulty (N=0, N=1, N=2). A value for N is picked randomly without replacement whenever a block segment is initiated. When all difficulties have been picked once, the next block begins. Block segments are structured as illustrated in figure 3.1. It begins with two seconds of segment presentation, where the subject is made aware of the segment difficulty. Then, nine series of screens are presented, each showing a stimulus from the set $\{C, F, H, S\}$ for 500 milliseconds, followed by a 1500 milliseconds long blank screen. Finally, a fixation cross is presented for six seconds. The idle, onset, execution, and offset labels in the figure will be detailed in section 3.4.2.

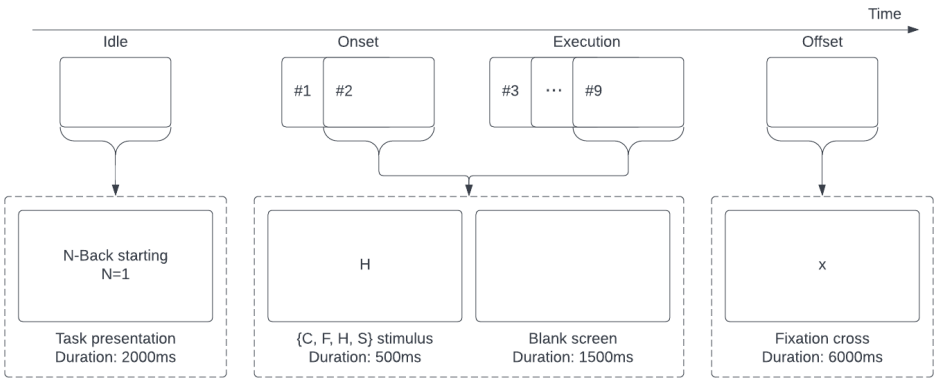


Figure 3.1: Timeline for each N-Back block segment. The current example represents N=1. One N-Back block represents three such block segments, one for each difficulty level.

3.3 Experimental Setup

3.3.1 Environment

The N-Back experiment was conducted at the Language Acquisition and Language Processing Lab at the Department of Language and Literature at NTNU. This lab hosts advanced equipment and facilities for research in experimental linguistics, psycholinguistics, and neurolinguistics. In addition to two EEG systems and an fNIRS system, they offer a variety of advanced research-grade eye trackers, one of which will be detailed in the subsection below.

The eye-tracking environment was fully enclosed to avoid outside disturbance during task operation. It was moderately lit to avoid interference in pupil diameter from the PLR. The screen which displayed the task stimulus was placed at a constant 65cm from the subject, ensuring minimal interference from the PNR as

well. The reason for such considerations were explained in section 2.2.1.

Recording sessions were limited to approximately five minutes (three N-Back blocks, each lasting about one and a half minutes), with at least two minutes of relaxation between sessions. This was done to mitigate the effects of fatigue on task performance. Additionally, since the author of this thesis was also the experimental subject, he was encouraged to be fully engaged in every task to ensure optimal data quality.

3.3.2 Hardware

The hardware used to record eye-tracking data was a Tobii Pro Spectrum, in a setup shown in image 3.2. This tracker has the advantage of having a permanently mounted stimulus screen. This setup minimizes error from poorly calibrated manual screen configurations, which is often the case with other models. Said stimulus screen was 23.8-inches wide and LED-backlit, with an aspect ratio of 16:9 and 1080p resolution.

Figure 3.2: Hardware setup of a Tobii Pro Spectrum in the Language Acquisition and Language Processing Lab.

The eye-tracker itself was mounted directly beneath the screen. It uses the pupil- and corneal reflection method detailed in section 2.5 to capture data in a non-intrusive manner. Its remote capabilities allow for free head movement within a 34cm x 26cm (width x height) track box, at 60cm to 80cm from the stimulus screen. Two cameras that capture stereo images of both eyes combined with nine infrared illuminators provide accurate estimations of on-screen gaze, 3D head position, and pupil diameter. All this data is streamed at up to 1200 samples per second, allowing for the capture of high-fidelity eye movements, tiny deviations in

pupil diameter, and high-frequency oscillatory eye movements such as microsaccades.

On-screen coordinates as output from the eye-tracker are depicted in figure 3.3. They are given in floating-point values from (0.0, 0.0) in the top-left corner to (1.0, 1.0) in the bottom-right corner. Upon installation, between subjects and otherwise, as often as possible, a calibration is required to get accurate gaze estimations.

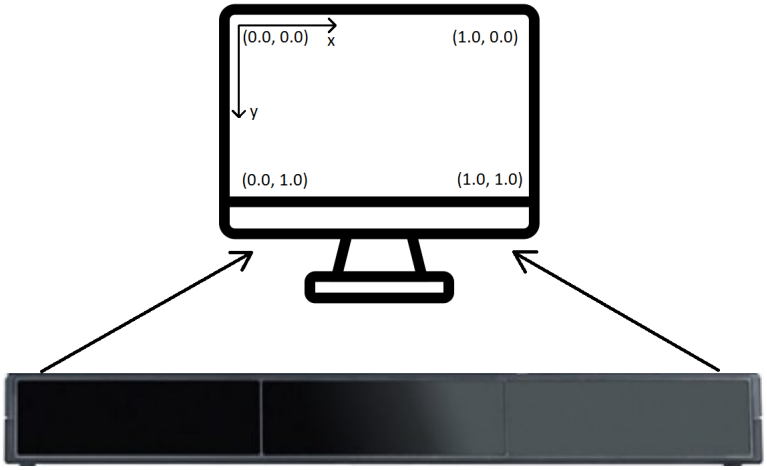


Figure 3.3: Schematic representation of the coordinate system in which on-screen gaze data is presented by the Tobii Pro Spectrum.

3.4 Dataset Creation

Raw data from the eye tracker was output in 16 channels. When data from one source was unavailable due to pupil occlusion or technical issues, the corresponding channel was filled with NaN-values. The raw data channels are presented in table 3.1. This table also gives a short description of each channel and the rate of missing data.

3.4.1 Pre-Processing

Raw data was not immediately suitable as a training dataset. Since many channels contained a high rate of missing values, these had to either be interpolated or dropped. Then, blink events and EBR could be extrapolated and added as a new channel.

Table 3.1: Description of the 16 raw data channels as output from the eye tracker.

Column name	Description	Missing data
Recording timestamp [us]	Time since start of recording in us	
Computer timestamp [us]	Unix time of host computer	
Pupil diameter left [mm]	Raw pupil diameter of left eye in mm	18%
Pupil diameter right [mm]	Raw pupil diameter of right eye in mm	15%
Gaze point X [MCS norm]	Left/right eye average on-screen gaze coordinate in X-dimension	14%
Gaze point Y [MCS norm]	Left/right eye average on-screen gaze coordinate in Y-dimension	14%
Gaze point left X [MCS norm]	Left eye on-screen gaze coordinate in X-dimension	19%
Gaze point left Y [MCS norm]	Left eye on-screen gaze coordinate in Y-dimension	19%
Gaze point right X [MCS norm]	Right eye on-screen gaze coordinate in X-dimension	15%
Gaze point right Y [MCS norm]	Right eye on-screen gaze coordinate in Y-dimension	15%
Gaze event duration [ms]	Total duration of present eye movement type in ms	
Sensor	Identifier of eye-tracker used for recording	
Validity left	True if data from left eye is available, else false	
Validity right	True if data from right eye is available, else false	
Presented Stimulus name	Label to which the current sample belongs	
Eye movement type	Type of eye movement, e.g. fixation or saccade	

Data cleaning

The first step of the data cleaning process was to remove all samples recorded before the session began. These samples were easily distinguished by their lack of a "Presented Stimulus name." Then, insignificant one- and two-sample data losses were interpolated with the last valid data point. Finally, a new data channel was added and filled with the average pupil diameter between the left and right eye.

Blink Extrapolation

Blinks were detected by applying algorithm 1. Its parameters were tuned by trial and error while observing the dataset's internal characteristics. Lines 2-6 looks for long-duration "EyesNotFound" eye movements and stores their indices in memory. Then, lines 8-21 further extend these blink intervals if short-duration "Unclassified" eye movements come within ten samples following the blink event. Finally, lines 23-27 remove blink-contaminated samples from the raw input data and create a new channel with EBR data.

Smoothing and Normalizing

A low-pass filter was applied to all data channels after the blink rate channel had been added. Then, the pupillary- and blink rate channels were scaled by minimum and maximum values to normalized levels between zero and one.

3.4.2 Labeling

After pre-processing, the data could finally be used to create a training dataset. To that end, the author chose four sets of labels, each generating one dataset intended to capture one inherent correlation within the data.

Levels

The first two datasets were labeled by the predefined difficulty levels of the N-Back task, as described in section 3.2. By the assumptions made in section 3.1, the labels of this dataset are intended to capture the levels of cognitive load imposed on the subject. Since each difficulty occurs the same number of times throughout the data, the labeling scheme ensures a fully balanced dataset. Furthermore, greater correlations may artificially be induced if the dataset is split into only levels 0 and 2. Throughout the following discussions, these datasets will be referred to as Full Level (FL) and Binary Level (BL).

States

In addition to the level datasets, another labeling scheme was employed. This scheme was intended to distinguish between the N-back task's transient states during execution. As was briefly mentioned in section 3.2.2, figure 3.1 has labeled each N-Back block segment with "idle", "onset", "execution", and "offset". As discussed in section 2.2, some cognitive states may be detected by task-evoked pupillary and EBR responses. This labeling scheme captured these correlations. Again, another dataset with fewer labels was added to induce larger correlations, with only the "idle" and "execution" states. We call these datasets Full State (FS) and Binary State (BS).

Algorithm 1 Blink extrapolation algorithm. EMT and GED is short for "Eye Movement Types" and "Gaze Event Durations", and represent two channels from the raw data with the same name. Note that the iterative representation of this algoritm is just for ease of understanding. In practice, it was implemented with efficient inplace data processing libraries.

```
1:  procedure EXTRAPOLATE(data)
2:      for sample in data do
3:          if sample.EMT = "EyesNotFound" and sample.GED > 5 then
4:              BlinkIndices ← data.index
5:          end if
6:      end for
7:
8:      for i ← 1 to 3 do
9:          for sample in data do
10:             if sample.index − 10 in BlinkIndices then
11:                 if sample.EMT = "Unclassified" and sample.GED < 2 then
12:                     BlinkIndices ← data.index
13:                 end if
14:             end if
15:         end for
16:         for index in BlinkIndices do
17:             if num. samples since last index > 10 then
18:                 BlinkIndices ← all indices since last index
19:             end if
20:         end for
21:     end for
22:
23:     data ← data − (data where data.index = BlinkIndices)
24:
25:     for sample in data do
26:         BlinkRate ← num. BlinkIndices in a window surrounding sample
27:     end for
28:
29:     return BlinkRate
30: end procedure
```

3.4.3 Segmentation, Subsampling, and Augmentation

Finally, the processed and labeled datasets were molded for Pytorch's tensor-based deep learning framework by arranging samples in fixed-length segments. Since data was recorded at frequencies beyond what was needed, these segments could be created by subsampling every fourth sample of the original data. This way, the dataset could be augmented from one hour of 1200Hz data to four hours of 300Hz data. Segments were chosen to be 256 samples in length, which could represent about 50s in real-time. Each sample was then assigned a *segment ID*. The target labels for each segment were added to a separate dataset.

The final dataset includes four data channels, as shown in table 3.2. They are pupil diameter, blink rate (EBR), and on-screen gaze in the X- and Y- dimensions. A fifth channel indicates which segment each sample belongs to, which determines its target label for classification.

Table 3.2: Description of the final five- and two-channel datasets after processing and labeling. The left table desribe the dataset that contains actual sample data. The right table only hold target labels for each segment.

Column name	Description
segment	Sample's segment ID
v_p	Left/right eye averaged pupil diameter, min/max normalized between 0 and 1
v_br	Blink rate, min/max normalized between 0 and 1
v_x	Left/right eye averaged on-screen gaze coordinate in X-dimension
v_y	Left/right eye averaged on-screen gaze coordinate in Y-dimension

Column name	Description
segment	Sample's segment ID
target	Target label for segment

3.5 Classification Models

All classification models detailed below were implemented in Python using toolsets provided by the Pytorch Deep Learning Library [1]. This allows for higher abstraction when designing model architectures without having to manually implement backpropagation, optimizers, activation functions, and other standard machine learning concepts. Leaving these implementations to the experts allows for GPU accelerated training optimized for computing efficiency.

While some models could be trained on a mere laptop CPU, all the deeper architectures were trained using a cloud computing instance provided by Amazon Web Services. The instance used had four Intel Xeon Cascade Lake P-8259L processors, 16GiB memory, and an Nvidia T4 Tensor Core GPU with 16GiB graphics memory.

3.5.1 Architecture

The architecture of a neural network defines the hypothesis space in which predictions may be made from a given input. Deeper networks can encapsulate more complex dataset features at the increased risk of overfitting features that do not generalize well outside the training data. For this thesis, the selection of model architectures is inspired by a review of deep learning of TSC algorithms presented by Fawaz *et al.* [76]. Although an overview is provided in the following subsections, a comprehensive list of all model architectures can be studied in appendix A.

All models aim to be fully end-to-end and able to classify multivariate time series of undefined dimensions. Input time series are multivariate in their data channels (e.g., pupillary, coordinate, and EBR) and temporal. The dimensions of the input data are therefore defined by segment length and the number of data channels.

Multi-Layer Perceptron

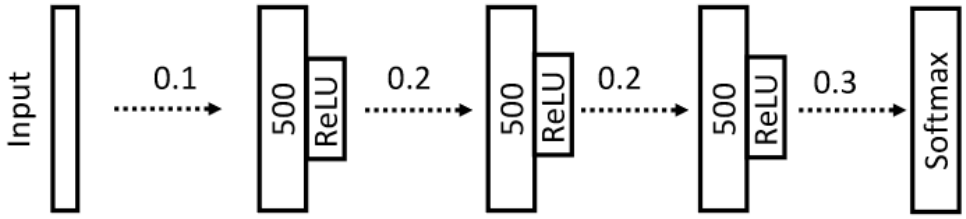


Figure 3.4: Multi-Layer Perceptron architecture.

Source: Wang *et al.* [77]

First up is the traditional and well-known Multi-Layer Perceptron (MLP). Being an integral part of deep learning, it is a natural inclusion as a baseline by which other architectures may be compared. This particular implementation, first proposed for TSC by Wang *et al.* [77], is presented in figure 3.4. It is composed of four layers where all neurons are connected to activations from neurons of the previous layer. The three hidden layers between input and output are each composed of 500 neurons with ReLU as the activation function. Between all layers is a dropout operation with dropout rates ranging from 10% to 30%. The input layer takes all samples in one time segment from all data channels. With input segments of length 256, the layers of this model contain $1,015,504$ parameters. Outputs from the last hidden layer are passed through a softmax function, such that the model output is a probability distribution over all classes that sum to one.

Since all layers in an MLP are fully connected, any temporal invariance between time-consecutive samples in the data is lost. This is because all input neurons to every layer are weighted independently from one another, such that the time dimension from input samples is essentially ignored. Since temporal invariance is an important factor for TSC, the MLP is likely a sub-optimal alternative to other deep learning architectures.

Fully Convolutional Network

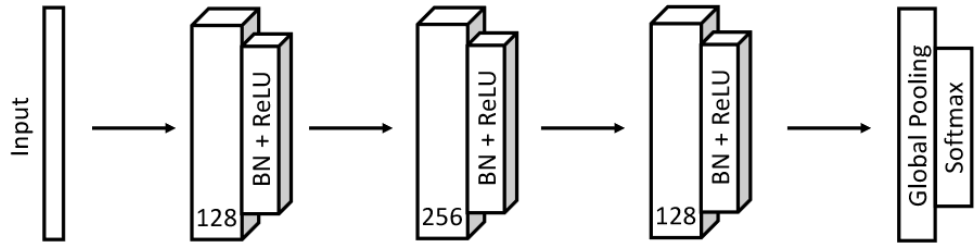


Figure 3.5: Fully Convolutional Network architecture.

Source: Wang *et al.* [77]

The second architecture presented is a form for CNN, as described in section 2.4.2. Like the MLP, the Fully Convolutional Network (FCN) implementation was proposed by Wang *et al.* [77], and is presented in figure 3.5. It is composed of three convolutional blocks, where each block is appended with a batch normalization layer which is passed through a ReLU activation function. Layers have 128, 256, and 128 convolutional filters, with filter lengths of eight, five, and three, respectively. Even though this architecture has the same layer depth as the MLP, the shared parameter- and sparse connection features of CNNs allows for fewer total parameters, at $268,292$. Output from the last layer is fed into a global average pooling layer that collapses the entire time dimension to a single value for every filter. These values are subsequently passed onto a fully connected classifier activated by a softmax function. Similar to the MLP, the output is, therefore, a probability distribution that sums to one.

Contrary to a MLP architecture, the convolutional nature of the FCN allow for temporal invariance. With this, the resulting model can recognize distinctive features in time. Another advantage of the FCN is invariance in the total number of parameters. Since the global average pooling layer collapses the time dimension, parameters are only dependent on the number of layers and filters, which is constant. This invariance is a consequence of the shared parameters property of CNNs. It allows for a transfer learning approach since the convolutional filter weights will learn features independent of segment length and temporal position.

Multi-Channel Deep Convolutional Neural Network

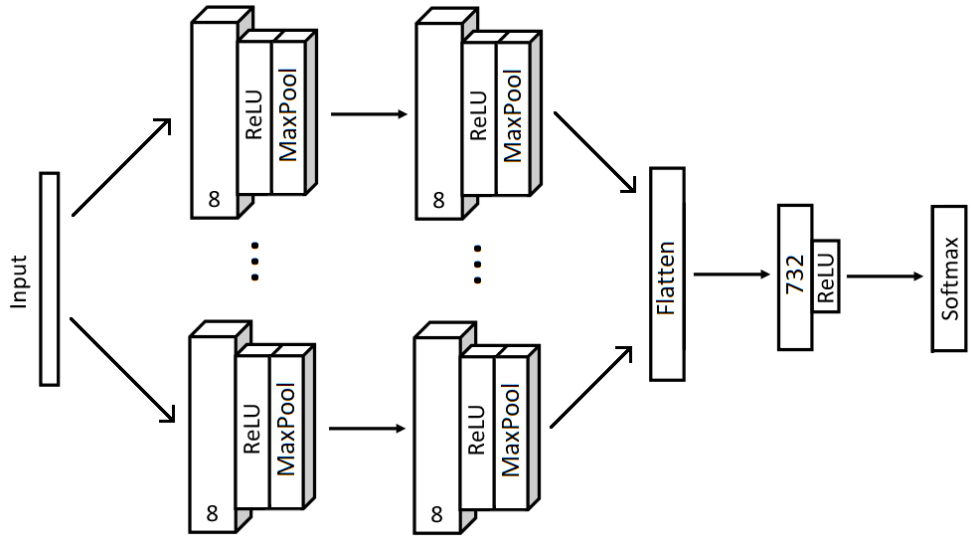
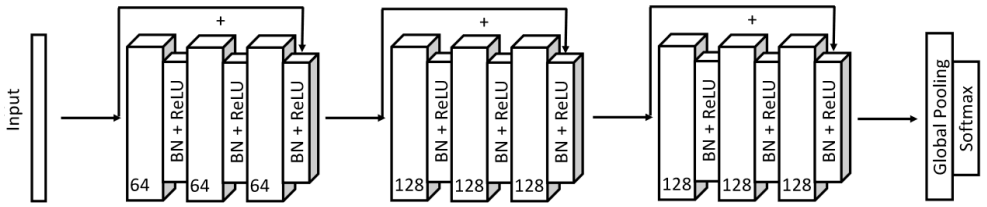


Figure 3.6: Multi-Channel Deep Convolutional Neural Network architecture.

The Multi-Channel Deep Convolutional Neural Network (MCDCNN) is in some ways a blend of a MLP and a FCN. It was originally proposed and validated by Zheng *et al.* [78], and was expected to perform especially well on multivariate time series datasets. As is illustrated in figure 3.6, it consists of two convolutional stages with eight filters of length five. Following the convolutions is a ReLU activation function and a max-pooling operation. Uniquely, these convolutional stages are all applied in parallel before their outputs are flattened and passed to a fully connected layer with 732 neurons, again with ReLU as its activation function. This model contains slightly more parameters than the FCN, at 378,632. Output from the fully connected layer is equal to the number of classes to be classified and is activated by a softmax function such that these class probabilities sum to one.

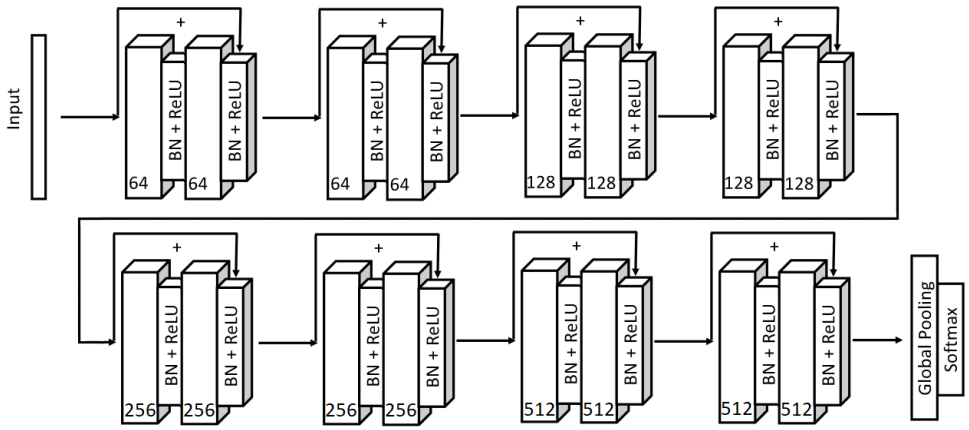
Separating input channels in individual parallel convolutions may have both advantages and disadvantages. On the one hand, the MCDCNN may be able to generate features that might otherwise have been lost from the cross-channel convolutions of the FCN. However, on the other hand, there might be interesting relationships across channels that may not be captured with this approach.

Residual Network



(a) 11-Layer Residual Network architecture.

Source: Wang *et al.* [77]



(b) 18-Layer Residual Network architecture.

Figure 3.7: Model architectures of ResNet11 and ResNet18. Three dimensional layers represent convolutions, numbered by filter count. Characteristic to ResNet architectures is the "skip-connections" represented by arrows skipping every two- or three convolutional blocks.

The two final models presented in this thesis are based on an architecture made famous for computer vision by He *et al.* [79], known as Residual Networks. At the time, a complication that was known as "the vanishing gradient problem" plagued most architectures of any considerable depth. It was caused due to the iterative training approach, where weights of the network are updated according to the gradient of the error function with respect to each weight. For every added layer, these values are multiplied, eventually approaching zero. In the worst case, this may slow and even halt the training process entirely [80]. Residual Networks solve this problem by introducing *skip connections* that allow for gradients to flow past convolutions from start to end. This architecture won several image classification competitions after its inception in 2015, and skip connections have been employed in many other state-of-the-art model architectures since.

In the original paper, He *et al.* [79] introduced five distinct ResNet architectures, ranging from 18 to 152 layers in depth. However, to avoid comparing models with over an order of magnitude of difference in depth, the architecture used

here will be limited to an 18-Layer Residual Network (ResNet18). This architecture is illustrated in figure 3.7b and employs eight *ResNet blocks*, each including two convolutional layers followed by a batch normalization layer activated by ReLU. The input to every ResNet block is saved and added to its output, which constitutes a skip connection. Internal convolutional layers use an increasing amount of filters over the depth of the network, from 64 to 512 filters. Just like the FCN, the outputs from the convolutional filters are sent through a global average pooling which collapses their time dimension. Finally, a fully-connected layer activated by softmax ensures a probability distribution over classes that sums to one.

A modification to the original ResNet architectures was proposed by Wang *et al.* [77]. This version, an 11-Layer Residual Network (ResNet11), has proven to be accurate on TSC, and is presented in figure 3.7a. Contrary to ResNet18, this version consists of three larger ResNet blocks, each with three convolutional layers employing 64, 128, and 128 filters. Otherwise, the input, output, and skip connections of this architecture are similar to that of ResNet18.

Both ResNet architectures sport the greatest of parameters of all convolutional models detailed and developed in this thesis. Their complexity is expected to influence the training process significantly. ResNet11 has $522,756$ parameters, while the deeper and more complex ResNet18 sports as much as $4,190,084$.

Chapter 4

Results

4.1 Dataset Visualization

4.1.1 Label groups

Figures 4.1 and 4.2 were created from the final dataset as processed according to section 3.4. However, instead of using the data as input for classification, it is segmented into the level and state label groups and plotted over time-on-trial. As such, each of the green-, blue-, red- and magenta-colored graphs in these figures represent average values over 43 trials. The same number of trials were used to generate the blue-colored histogram above each plot. Each bar in the histogram represents one 50ms interval. Their heights are normalized according to the number of blinks during each interval.

As is evident from their titles, each row of plots represents one difficulty level of the N-Back task. Additionally, all are marked with black vertical lines along the time axis. These are meant to indicate the points in time where the task transitions from one state to another, as described in section 3.4. The respective state at each time instance is indicated by the "idle," "onset," "execution," and "offset" labels at the top of each plot.

4.1.2 Task Exposure

Figure 4.3 was generated similarly to figures 4.1 and 4.2. However, this figure demonstrates how continuous task exposure affects the data. Plot rows each represent one data channel, as is evident from their titles. In each plot, graphs are colored by which time during task exposure they were recorded. For instance, the green graph comes exclusively from the first quarter of the recording session (first 15 minutes). The blue comes from the second quarter (15-30 minutes), and so on. These plots are also indicated with vertical lines representing task states, similar to the above figures.

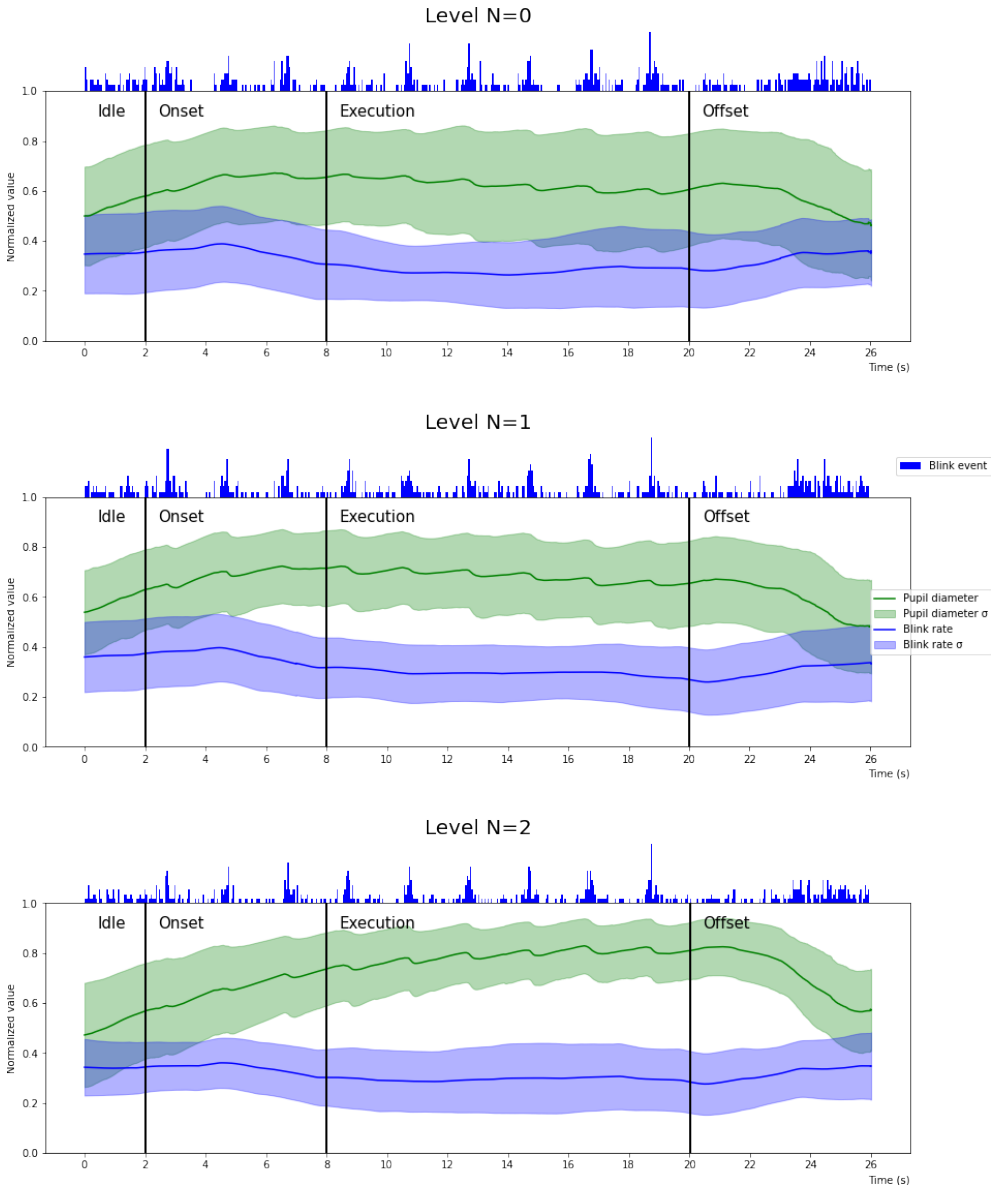


Figure 4.1: Visualization of state and level labels of the processed dataset. The green and blue graphs represent the pupillary diameter and blink rate data channels, respectively. The blue histogram represents individual blink occurrences during 50ms intervals throughout the trial. Black vertical lines indicate task state transitions.

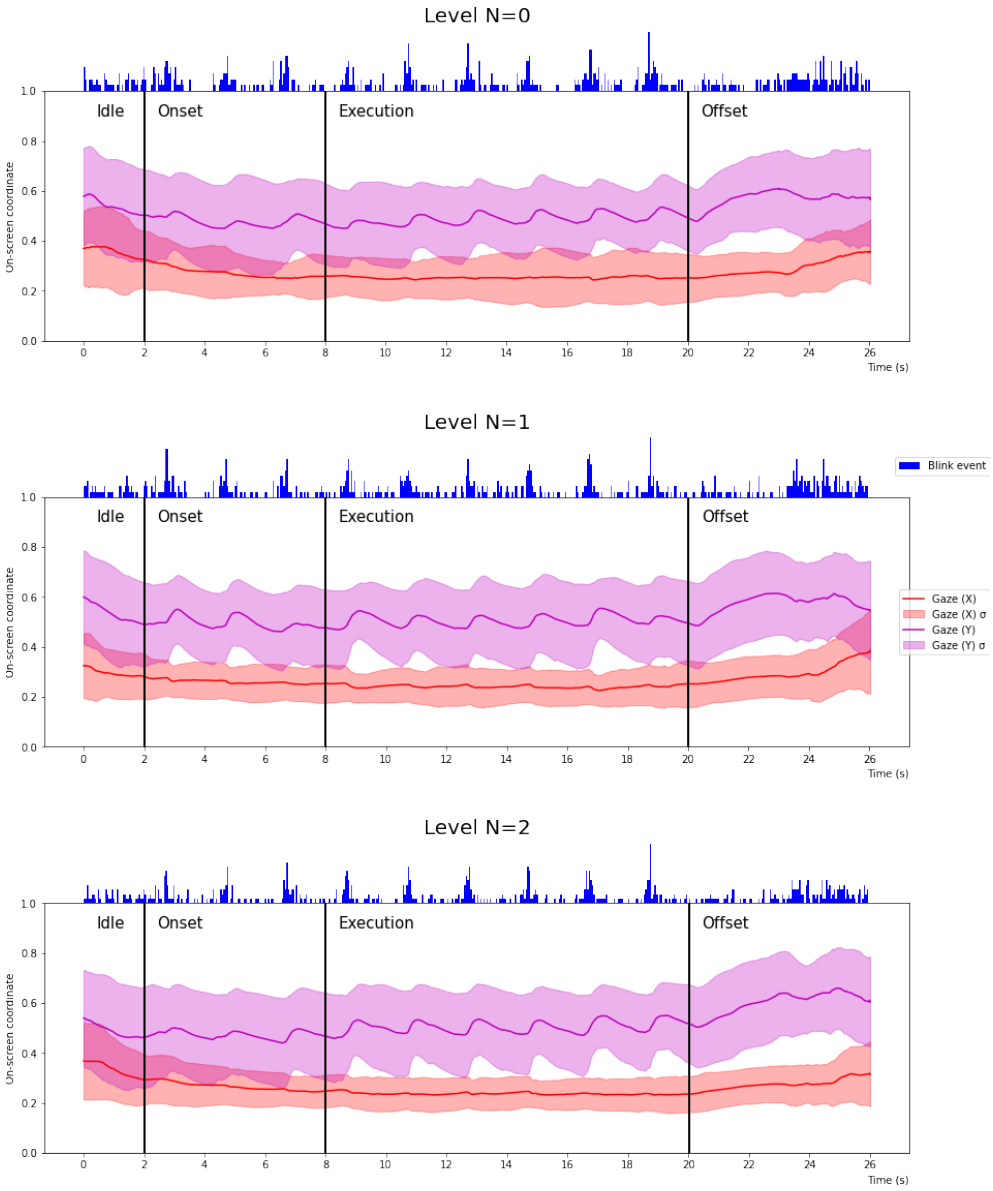


Figure 4.2: Visualization of state and level labels of the processed dataset. The magenta and red graphs represent the on-screen x- and y-coordinate data channels, respectively. The blue histogram and black vertical lines are similar to figure 4.1.

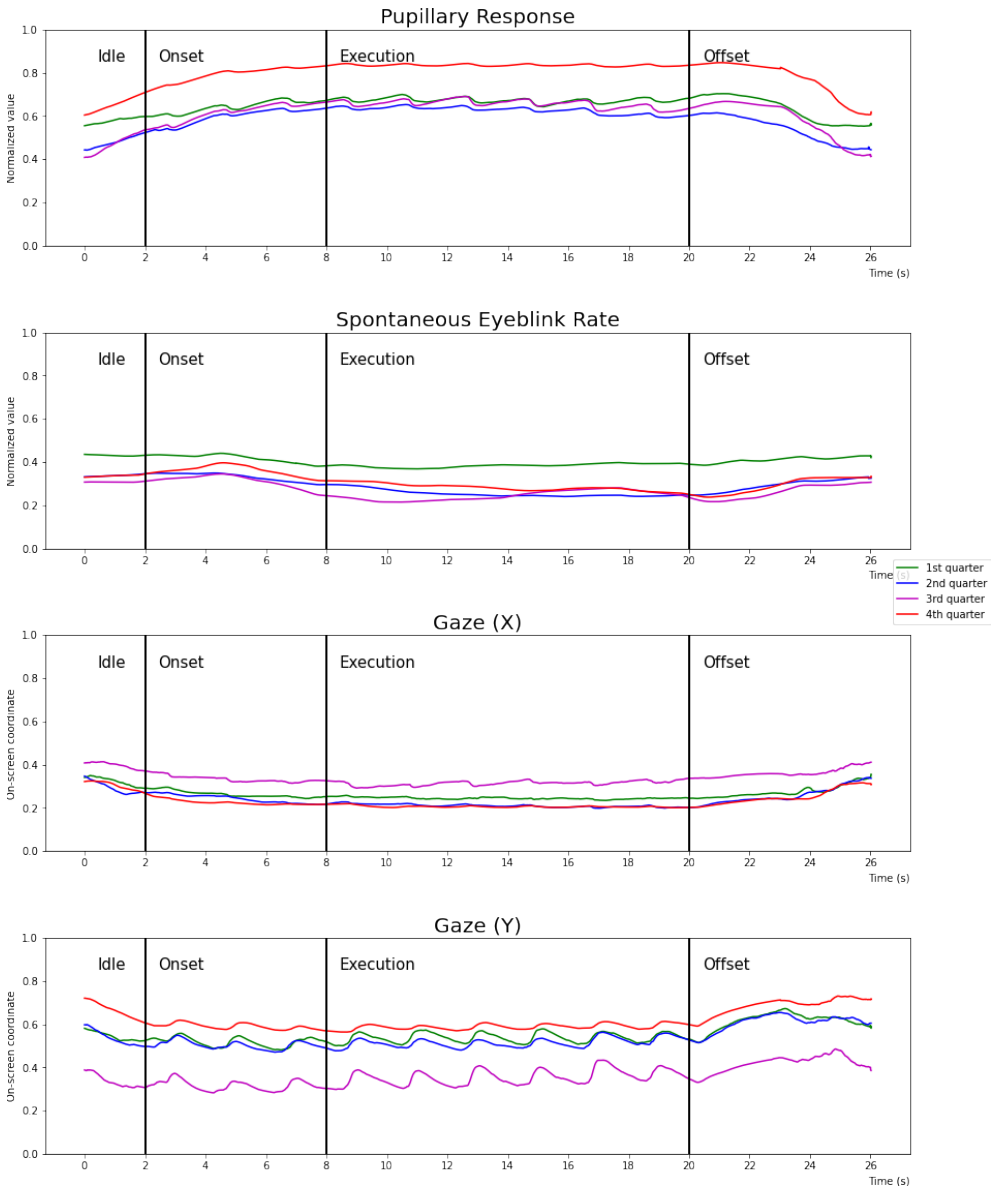


Figure 4.3: Visualization of task exposure effect in dataset. Graphs are color-coded by which quarter of task exposure they were recorded. The black vertical lines are similar to figure 4.1

4.2 Classification Accuracies

Table 4.1 gives an overall summary of the classification accuracies observed for the various model architectures discussed in section 3.5. Each model was trained on the four label groups detailed in section 3.4.2. Classification accuracy is calculated by passing a randomly selected set of input sequences through the model and observing its predicted class for each sequence. The proportion of correct predictions gives an accuracy score.

Table 4.1: Classification accuracy for all model architectures and label groups. The best model for each label group is highlighted in **bold**

Label \ Model		MLP	FCN	MCDCNN	ResNet11	ResNet18
State	Full	-	**71.43%**	-	70.78%	66.88%
	Binary	-	94.24%	-	**95.40%**	94.25%
Level	Full	52.78%	**61.11%**	48.61%	58.33%	58.32%
	Binary	65.62%	73.96%	69.79%	77.08%	**82.29%**

4.3 Training

All classification models were trained on the cloud computing instance mentioned in section 3.5. The time and computing resources required for training depended on label groups and the size of input tensors. However, results from averaging over these groups can be used to present an overall comparison between models. These results are given in table 4.2.

Table 4.2: Average time and epochs elapsed during training for all model architectures.

Average metrics \ Model	MLP	FCN	MCDCNN	ResNet11	ResNet18
Epochs until best result	711	555	150	262	304
Training time per epoch	0.5s	5.5s	1.7s	7.9s	42s
Total training time	11m 52s	50m 52s	4m 16s	41m 11s	3h 33m

4.3.1 History

The following plots present the history of training loss and accuracy for the two best models on the FS dataset; FCN and ResNet11. Training loss was captured by logging the cross-entropy loss used to train model parameters during backpropagation. Validation accuracy was calculated by running the model on a validation dataset for every epoch. It is a percentage measure of how many classes were predicted correctly from a given set of inputs.

Fully Convolutional Network

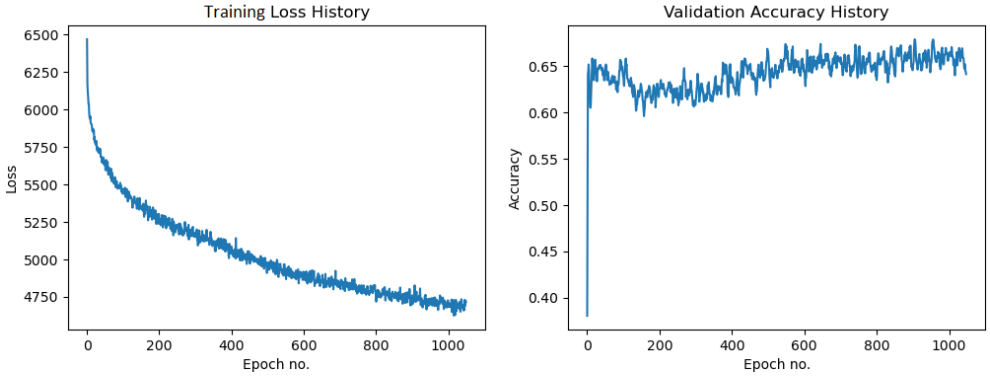


Figure 4.4: Training history for Fully Convolutional Network on the FS label group. Left graph shows the cross-entropy loss for every epoch. Right graph shows classification accuracy on the validation dataset for every epoch.

11-Layer Residual Network

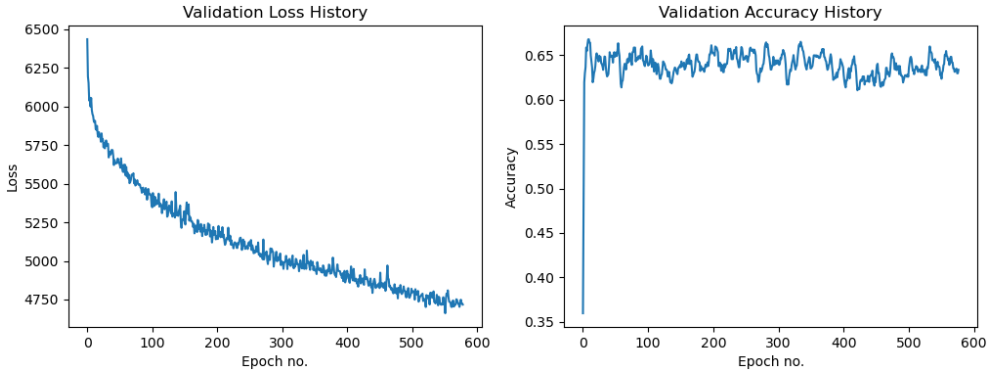
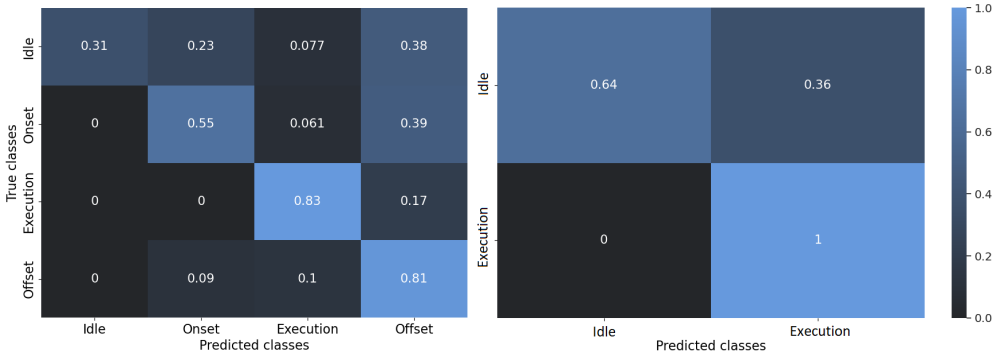


Figure 4.5: Training history for 11-Layer Residual Networkon the FS label group. See figure 4.4 for plot description.

4.4 Confusion Matrices

The following subsections will present a visualization of the predictions made by the two best models for every label group in table 4.1. Visualizations are plotted from the confusion matrices from which the classification accuracies were calculated. Therefore, the following visualizations will display one row and one column for every class in a label group. Each input sequence is passed through the model and added to one cell, whose row and column correspond to its correct and predicted class. The counts in every cell are finally normalized for every row. The resulting values in each cell can therefore be interpreted as the "probability that the model will predict [column class] given [row class] input," a metric more commonly known as *recall*. A perfect classification model will only have one's along the diagonal. Off-diagonal values represent misclassifications.

4.4.1 State Label Groups

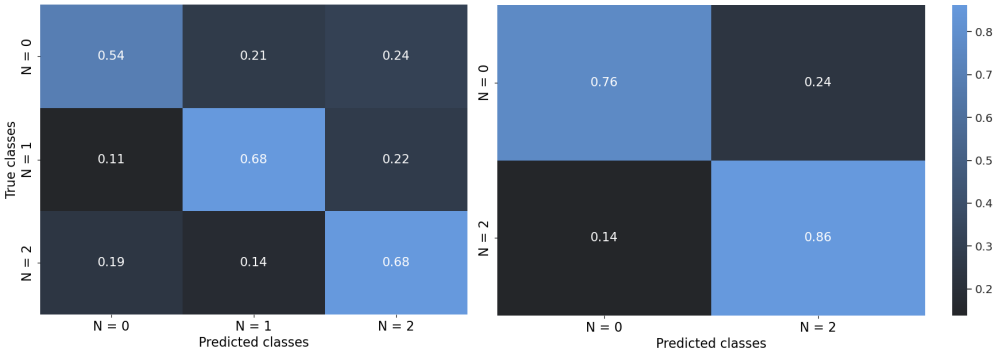


(a) FS dataset classified by Fully Convolutional Network

(b) BS dataset classified by 11-Layer Residual Network

Figure 4.6: Confusion matrices for thestate label groups detailed in section 3.4. Cell values and color gradients indicate proportion of input segments classified as [column], with true class [row].

4.4.2 Level Label Groups



(a) FL dataset classified by Fully Convolutional Network

(b) BL dataset classified by 18-Layer Residual Network

Figure 4.7: Confusion matrices for the level label groups detailed in section 3.4. Cell values and color gradients indicate proportion of input segments are similar to figure 4.6.

Chapter 5

Discussion

5.1 Dataset Properties

Creating a processed dataset ready for analysis from a raw eye-tracking data stream turned out to be no trivial task. The high throughput of the 1200Hz eye-tracker made high-quality data abundant. However, data anomalies and noisy samples became just as problematic. Consequently, data processing was a significant part of the implementation process, entirely dwarfing other areas in terms of time invested. As detailed in section 3.4, subject blinking was the largest source of problems since the tracker continued recording empty data with closed eyes. Even after the gaze had been recovered following the blink event, hundreds of subsequent samples were contaminated by large amounts of noise. A representation of how the final dataset turned out is illustrated in figures 4.1, 4.2, and 4.3.

As the blue histograms above the three graphs of figures 4.1 and 4.2 show, blinks seem to be remarkably consistent throughout every N-Back block segment. Blink occurrences are so consistent that the data anomalies they cause, as mentioned above, are apparent in visualizations. Pupil diameter in figure 4.1, as well as on-screen gaze on the y-axis of figure 4.2 was particularly affected by this. These graphs' sawtooth pattern suggests that a rapid data disturbance is caused after every blink occurrence, followed by a gradual rebound to pre-blink levels.

Both the effect on gaze and pupil diameter may be explained by a limitation with the pupil- and corneal reflection method described in section 2.5. Since this algorithm relies on the outline of the pupil to determine both the pupil center and its diameter, it is natural to believe that occlusion of the pupil caused by closed eyelids will cause a disturbance. This effect is worth keeping in mind since a classification model is likely to pick up such features when making predictions. Since it is only caused by an anomaly and has no grounds in physiology, a model trained on such patterns is likely to generalize very poorly to unseen data.

5.2 N-Back and Cognitive Load

The fact that the levels and states of the N-Back task are a sufficient proxy for cognitive load is a foundational pillar on which this entire thesis builds. If this were not the case, the classification models developed would have no value besides predicting various time intervals of a monotonous and uninteresting attention experiment. As detailed in section 3.1, there are mainly five assumptions that need to hold for this pillar to stand.

Since the author was also the subject in all trials, he may make some subjective assessments on assumptions 1 and 2. First, assumption 2 holds since the author was positively motivated to be actively engaged throughout the trials. Second, there can be no doubt that the task levels required distinctly different demands on working memory. Level 0 was trivial. After just a few stimulus screens, responding became entirely autonomous. Level 1 did require some sustained effort, and it was no longer possible to react correctly without actively engaging in the task. Level 2, however, was remarkably demanding. The subject required many consecutive trials to respond correctly to more than half of the stimuli. Therefore, we can confidently conclude that task levels sufficiently reflect the load on working memory. Despite this, it is worth noting that the load imposed was not necessarily linear with the difficulty level. While level 1 was slightly more challenging than level 0, level 2 was substantially more demanding than both the lower levels. This limitation is addressed by introducing the binary level label group, which only distinguishes between the least and most demanding levels.

Assumptions 3 and 5 are argued for in section 2.1.1 and 2.2.1, respectively. In short, the intensity of effort expended on a task (mental effort) is considered the essence of cognitive load [19] because other metrics are unreliable and subjective. Additionally, the effects of the PNR and PLR may easily be accounted for by carefully controlling the recording environment. This includes the lighting and monitor position.

Finally, the potential uncertainties highlighted by assumption 4 is visualized in figure 4.3. For instance, we can see from the pupillary response of the top plot that data collected from the final quarter of task exposure deviates significantly from the prior three quarters. As discussed in section 2.2, the pupil dilating and constricting fibers are tightly coupled with the central nervous system. Pupil dilation, in particular, is known to be triggered by the fight-or-flight response and the stimulus of the PNS. Therefore, what we see in the final few minutes of recording may just be attributed to impatience or fatigue. This result is unexpected given the experiment by Hopstaken *et al.* [36], reproduced in figure 2.4b. They found the opposite relation; baseline pupil diameter decreased with time-on-task. The fact that our results conflict may indicate a lack of sufficient data quantity and quality. After all, only data from one subject in one session was recorded.

The eyeblink rate of the second plot in the same figure shows another interesting relationship. Here, the first quarter of recording indicates slightly elevated values compared to the rest of the session. As we also know from section 2.2, EBR

is known to be affected by the assertion of cognitive control. The subject required a few consecutive trials of the hardest difficulty before responding correctly to most stimuli. This "learning phase" may have induced heightened levels of cognitive control, which may, in turn, have affected EBR.

5.3 Ocular Correlations with Cognitive Load

As the author spent several pages exploring in section 2.2, the literature shows that there should be a clear correlation between the inner workings of the mind and various ocular events. With the dataset made available for this thesis, we can open the discussion on how these correlations play out in practice.

5.3.1 Pupillometry

The pupillary response throughout each N-Back block segment can be observed by the green graph in figure 4.1. First of all, values during the onset and offset task states show a striking resemblance with the results found by Kahneman and Beatty [35], reproduced in figure 2.4a. They attributed pupil dilation on task onset to the increased number of digits that had to be kept in working memory at any one time. The pupil then constricted to pre-task levels after digits were unloaded. Similarly, the N-Back task requires the active engagement of working memory by memorizing a target stimulus. When transitioning from the idle state and into the first few stimulus screens, the subject is required to keep at least one letter memorized. Subsequently, when the subject was finally allowed to relax working memory by forgetting the target, the pupil again constricted to baseline levels.

The fact that the effect is more pronounced for level 2 than level 0 further couples with the results by Kahneman and Beatty [35], which showed the same relation for long digit series as opposed to short. For one, figure 4.1 show an average pupil diameter that is slightly higher for level 2. Even more evident, however, is its consistency. This consistency can be seen from the area drawn by the green graph's standard deviation. Especially during the execution state, level 2 pupil diameter shows consistent values above 0.7, with standard deviations closer to 0.1. Compared to average values of about 0.6 with standard deviations around 0.2 for level 0, the difference is significant. It seems clear from these results that phasic pupil dilation reflects working memory activation. Furthermore, the amount of information processed manifests itself in the magnitude of dilation.

5.3.2 Spontaneous Eyeblink Rate

Both the blue histogram and the blue graphs of figure 4.1 are an indicator of EBR. The distinct groups of intervals in which blink events occur are of particular interest. The subject blinks about six times at a remarkably constant rate during the execution state. The groups are more discernible for level 2 than for level 0. Similar to the results found by Oh *et al.* [49], reproduced in figures 2.6 and 2.7,

this behavior may relate to the task-evoked EBR detailed in section 2.2.2. Oh *et al.* [49] argues that distinct blinking behaviors can be observed at the onset of a task, which may be just what we observe in the present results.

With the results by Oh *et al.* [49] in mind, one would expect to see an increase in baseline EBR during task execution. Yet, what we see from figure 4.1 is the opposite. Blink rate seems to decrease during task onset and increase during task offset. Perhaps this is the result of the N-Back task not inducing heightened levels of cognitive control as much as it induces cognitive load. As detailed in section 2.2.2, different experiments are known to correlate with different physiological measures. If, for example, the Stroop task had been chosen instead, we might have seen a stronger correlation with EBR.

However, the poor EBR correlation with task states also shed some light on a limitation with the present setup of the N-Back task. In the experiment by Oh *et al.* [49], the rest condition lasted for as long as two minutes. Contrary to the present task, where the subject was only allowed six seconds of offset plus two seconds of rest, there is reason to believe that the task periods were too short to observe some of the inherent trends in the physiological responses. The pupillary data tell the same story. If the idle period was extended tenfold, the data quality and reliability would likely improve significantly.

5.3.3 Gaze

As explained in section 2.2.3, gaze patterns do not have striking evidence in the literature which confirms their correlation with cognition. However, as we can see from figure 4.2, there are distinct features within the gaze coordinates recorded throughout task execution. First, the same sawtooth pattern in the pupillary data is quite apparent in the on-screen Y-coordinates. This observation is natural since the occlusion of the upper part of the pupil will disturb the pupil- and corneal reflection algorithm when calculating gaze position.

Another observation is the slight reduction in variance during task execution for both X- and Y-coordinates. Interestingly, the variance reduction is most distinct in level 2 compared to 0. These results are consistent with what Williams [51] found in his study of a subject's field of view as a response to task-evoked cognitive load. Therefore, what we see here may be what he associates with a human tendency to subdue field of view when highly concentrated or under a heavy mental workload.

5.4 Classification

From tables 4.1 and 4.2, there is no denying that the architecture of a classification model matters. On the one hand, simpler models require less computing resources and time to train but are far surpassed by the more complex models in terms of classification accuracy. In the end, the confusion matrices of section 4.4

are what ultimately indicate how the classification models may be compared in a production setting.

Overall, it is clear that the CNNs are a winner for Time Series Classification. This result is similar to what Fawaz *et al.* [76] found in their review. Contrary to their results, however, FCN outperforms both ResNet11 and ResNet18 for two out of four datasets. These findings are surprising, considering that deeper networks are known to perform better than shallower networks for computer vision [79]. One reasonable justification would blame limitations in the dataset for this inconsistency. While deep models often produce better results than their shallow counterparts, they require much larger datasets to generalize well. This is natural, given that higher complexity raises the risk of overfitting the training data. In any case, while this outperformance was only marginal by one or two percentage points, ResNet18 dominated FCN by almost 12 percentage points when trained on the BL dataset.

The lack of temporal invariance in the MLP and MCDCNN architectures, as highlighted in section 3.5, was made clear when they were trained on the FS and BS datasets. Contrary to FL and BL, classes of the state label groups have more pronounced temporal distinctions in their input channels. For instance, as can be observed by figure 4.1, the onset state shows a positive slope pattern in the pupillary channel and a negative slope in the blink rate channel. Since the convolutional nature of FCN, ResNet11, and ResNet18 allow for the creation of feature maps that encapsulate patterns in time, they are naturally suited to perform well on such datasets. MLP and MCDCNN, however, were unable to improve their weights beyond making chance predictions. It is for this reason that some cells are excluded from table 4.1.

As expected, the deeper convolutional networks took much longer to train than MLP and MCDCNN, as can be seen from table 4.2. In fact, even though FCN had less than half the amount of parameters as MLP, it took over four times longer to train. Not surprisingly, this is caused by the parameter sharing and sparse interaction properties mentioned in section 2.4.2. Instead of learning individual weights for every input to every output node in every layer, convolutional networks learn internal weights of shared filters, each representing one feature in the input. However, training is still more computationally expensive since these filters must be convolved across the entire time dimension.

The two plots presented in section 4.3 tell a story of how quickly each model architecture saw improvements in classification accuracy during training. Observe that, although both the FCN and ResNet11 architectures gave about the same level of cross-entropy loss when training was stopped, ResNet11 reached these values at half the number of epochs. This result may be due to the depth and complexity of ResNet11, allowing it to make larger steps towards a local optimum for every epoch. The deeper ResNet11 architecture also overfits the training data much earlier than FCN, at around epoch number 300. After this point, the validation accuracy begins deteriorating while the validation loss keeps decreasing. FCN shows less tendency to overfit, reaching epoch 900 before the validation accuracy starts

deteriorating.

5.4.1 Confusion Matrices

Although accuracy is a good metric to evaluate a model's overall generalized performance for every training step, the confusion matrices of section 4.4 allow us to look closer at its ability to predict individual classes. All matrices have been normalized along their rows, such that row cells illustrate probability distributions that sum to one. Doing so aligns values such that the cells on the diagonal represent per-class recall. The same cells would instead represent per-class precision had values been normalized along columns. However, since we are interested in discussing the properties of true classes in the dataset, the recall metric was deemed reasonable for this thesis.

First, looking at the classifications of the state label groups in figure 4.6, we can see a clear pattern of correct predictions along the matrix diagonals. However, some segments are more distinctly recognizable in the data than others. The execution and offset classes are correctly recalled in over 80% of cases, while idle is mostly misclassified. Onset, too, is often misclassified as being offset. The characteristic sawtooth pattern of the execution class may explain how its classification performs so well. By looking at the gaze channels of figure 4.2, one may understand the discrepancy with the idle class by comparing some of the transients that it exhibits compared to the onset and offset classes. It is hard to recognize patterns that positively place a segment in the idle class even by human judgment. Combined with the fact that the state dataset is highly imbalanced (few examples of idle segments are presented to the model during training), poor performance in this class is to be expected.

By removing the onset and offset classes entirely, as in the BS dataset, we get the confusion matrix of figure 4.6b. Now, 33% of the previously misclassified idle segments add up to a recall of 64%. Although an improvement, it is not at all perfect. Again, the imbalanced nature of the state dataset may be to blame. Recall of the execution class is also greatly improved, up to an impressive 100%.

The results from level classification, presented in figure 4.7, show a similarly distinct pattern on the matrix diagonal. However, since classes of the level label groups are both fully balanced (each level is represented in 33% of all segments) and less temporally invariant, we see more of a spread in cell values. Interestingly, recall is almost 15% higher for levels 1 and 2 than for level 0. This result is reassuring for the case of cognitive load classification. It signifies that there is indeed a data pattern in these more cognitively demanding levels that the model picks up. In particular, the lack of consistency of data from level 0 compared to 1 and 2, mentioned in section 5.1, may explain why level 0 is more often misclassified.

Chapter 6

Conclusion

With this project, we have gotten to know the intimate relationship between our eyes and cognition. Pupil size and Spontaneous Eyeblink Rate are highly correlated with attentional focus and cognitive control. Other gaze-related aspects such as fixation duration and field-of-view are similarly associated with the mind. Section 2.1 made the argument that cognitive load may be used as a broad measure of these associations. With the advent of commercially available eye-tracking technology, these findings suggest that it should be possible to make accurate cognitive load measurements available for everyone.

To this end, we have studied a range of machine learning architectures, searching for an optimal model with which cognitive load may be classified. According to expectation, the deeper Convolutional Neural Networks came out on top, showing that fully connected networks are unsuited for multivariate Time Series Classification.

Supervised learning was conducted on a high-quality multivariate dataset, manually recorded in a setting and on a task that could isolate cognitive load as a ground truth. By visualizing this data by its internal levels and states, one could comfortably observe pupillary and gaze responses to cognitive load that matches the literature. EBR, however, behaved somewhat contrary to expectations, hinting at limitations faced by the task implementation. Despite this, the surprising consistency of blink events surrounding task state transitions still justifies its inclusion in the dataset. All data channels positively contributed to the classification accuracies observed by the various models explored.

The best models showed accuracies up to 71% when classifying four states of cognitive load and 61% when classifying three levels. Accuracies improved by about 20 percentage points in both cases when the problem was reduced to binary two-class distinction. Nonetheless, they lack generalization since they have only been trained on limited intra-subject and intra-task datasets. As discussed in section 2.6, such a deficiency completely devalues the model in a production setting, as ocular responses to cognitive load are both very subjective and task-dependant. The primary goal of this thesis was to create a model that could classify cognitive load from eye-tracking data with accuracies beyond chance. With this in mind,

the models developed outperformed expectations as proof of concept.

For Osirion, this work will lay the foundation for further research into cognitive load and time series classification. There are likely many untapped opportunities in the exploration of other data sources beyond just eye-tracking. One could imagine an application that captures such data together with eye movements, all in the background of everyday activities. With a sufficiently large model and sufficiently large labeled training sets, the development of a platform that may access the deepest of insights into personal health and growth may be within reach.

6.1 Future Work

Future work should emphasize improvements in the dataset. Generalizability should be addressed by introducing a larger participant group and task selection. A model trained on such data would be transferable to unseen subjects and tasks. Additionally, the dataset could be significantly improved by redesigning the recording environment, taking care of the weaknesses discussed in sections 5.3.2 and 5.4.1. For instance, modifying the task such that all states have equal length would allow for the pupillary and blink rate responses to fully settle between trials and address the problem of imbalanced classes.

The dataset may also be favorably expanded with more modes of data. Smartwatches, for instance, offer many physiological data sources, which have so far remained unexplored in this thesis. Heart rate, blood oxygen levels, respiratory rate, and skin conductance are not uncommon in modern models. Although perhaps not as prominent as pupillometry and EBR, there is much research on the correlation of such data with cognition [16, 81–84].

Bibliography

[1] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, Curran Associates, Inc., 2019, pp. 8024–8035.

[2] C. Ware and H. H. Mikaelian, "An evaluation of an eye tracker as a device for computer input2," 1986.

[3] T. v. Gog and H. Jarodzka, "Eye tracking as a tool to study and enhance cognitive and metacognitive processes in computer-based learning environments," in *International Handbook of Metacognition and Learning Technologies*, R. Azevedo and V. Aleven, Eds. New York, NY: Springer New York, 2013, pp. 143–156.

[4] P. Barry, J. Dockery, D. Littman, and M. Barry, "Intelligent Assistive Technologies," *Presence: Teleoperators and Virtual Environments*, 1994.

[5] F. Corno, L. Farinetti, and I. Signorile, "A cost-effective solution for eye-gaze assistive technology," in *Proceedings. IEEE International Conference on Multimedia and Expo*, 2002.

[6] J. D. Leyba and J. Malcolm, "Eye tracking as an aiming device in a computer game," 2004.

[7] J. D. Smith and T. C. N. Graham, "Use of eye movements for video game control," Hollywood, California, USA: Association for Computing Machinery, 2006.

[8] Tobii, *Eye tracking in gaming, how does it work?* Accessed: 2021-10-17, 2017.

[9] J. Antunes and P. F. Santana, "A study on the use of eye tracking to adapt gameplay and procedural content generation in first-person shooter games," *ArXiv*, 2018.

[10] E. Mangeloja, "Economics of esports," 2019.

[11] Newzoo, "Free 2018 global esports market report," 2021.

[12] E. Hess, *Attitude and Pupil Size* (Scientific American offprints). W.H. Freeman Company, 1965.

[13] D. Kahneman, *Thinking, fast and slow*, 2013.

[14] J. Sweller, J. J. G. Van Merrienboer, and F. Paas, "Cognitive architecture and instructional design," *Educational Psychology Review*, vol. 10, pp. 251–, Sep. 1998.

[15] J. Sweller, "Cognitive load during problem solving: Effects on learning," *Cognitive Science*, vol. 12, no. 2, pp. 257–285, 1988.

[16] F. Paas, J. J. G. Van Merrienboer, and J. Adam, "Measurement of cognitive load in instructional research," *Perceptual and motor skills*, vol. 79, pp. 419–30, Sep. 1994.

[17] F. Paas and J. J. G. van Merriënboer, "Variability of worked examples and transfer of geometrical problem-solving skills: A cognitive-load approach," *Journal of Educational Psychology*, vol. 86, pp. 122–133, 1994.

[18] M. K. Tulga and T. B. Sheridan, "Dynamic decisions and work load in multitask supervisory control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 10, no. 5, pp. 217–232, 1980.

[19] P. Hamilton, "Process entropy and cognitive control: Mental load in internalised thought processes," in *Mental Workload: Its Theory and Measurement*, N. Moray, Ed. Boston, MA: Springer US, 1979, pp. 289–297.

[20] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," in *Human Mental Workload*, ser. Advances in Psychology, P. A. Hancock and N. Meshkati, Eds., vol. 52, North-Holland, 1988, pp. 139–183.

[21] M. T. C. Team, *Cognitive load theory, helping people learn effectively*, Accessed: 2022-03-08, 2016.

[22] R. Klatzky and N. Giudice, "Sensory substitution of vision: Importance of perceptual and cognitive processing," in Jan. 2012, pp. 162–191.

[23] W. Wells, *An Essay Upon Single Vision with Two Eyes: Together with Experiments and Observations on Several Other Subjects in Optics* (Eighteenth century collections online). T. Cadell, in the Strand, 1792.

[24] M. K. Eckstein, B. Guerra-Carrillo, A. T. Miller Singley, and S. A. Bunge, "Beyond eye gaze: What else can eyetracking reveal about cognition and cognitive development?" *Developmental Cognitive Neuroscience*, vol. 25, pp. 69–91, 2017.

[25] F.-C. Huang and B. Barsky, "A framework for aberration compensated displays," Jan. 2011.

[26] J. Beatty, "Task-evoked pupillary responses, processing load, and the structure of processing resources.," *Psychological bulletin*, vol. 91 2, pp. 276–92, 1982.

[27] S. R. Steinhauer, G. J. Siegle, R. Condray, and M. Pless, "Sympathetic and parasympathetic innervation of pupillary dilation during sustained processing," *International Journal of Psychophysiology*, vol. 52, no. 1, pp. 77–86, 2004.

[28] M. A. Just, P. A. Carpenter, and A. Miyake, "Neuroindices of cognitive workload: Neuroimaging, pupillometric and event-related potential studies of brain work," *Theoretical Issues in Ergonomics Science*, vol. 4, no. 1-2, pp. 56–88, 2003.

[29] A. J.L., "Pupillary response and behavior," *Psychophysiology: Human Behavior and Physiological Response*, pp. 218–233, 2000.

[30] B. Blessing and I. Gibbins, "Autonomic nervous system," *Scholarpedia*, vol. 3, no. 7, p. 2787, 2008.

[31] J. Rajkowski, "Correlations between locus coeruleus (lc) neural activity, pupil diameter and behavior in monkey support a role of lc in attention," *Soc. Neurosc., Abstract, Washington, D. C., 1993*, 1993.

[32] S. Sara, "The locus coeruleus and noradrenergic function," *Nature reviews. Neuroscience*, vol. 10, pp. 211–23, Mar. 2009.

[33] D. Bylund and K. Bylund, "Norepinephrine," in *Encyclopedia of the Neurological Sciences (Second Edition)*, M. J. Aminoff and R. B. Daroff, Eds., Second Edition, Oxford: Academic Press, 2014, pp. 614–616.

[34] B. P. Ramos and A. F. T. Arnsten, "Adrenergic pharmacology and cognition: Focus on the prefrontal cortex.," *Pharmacology & therapeutics*, vol. 113 3, pp. 523–36, 2007.

[35] D. Kahneman and J. Beatty, "Pupil diameter and load on memory," *Science*, vol. 154, pp. 1583–1585, 1966.

[36] J. Hopstaken, D. Linden, and M. Kompier, "A multifaceted investigation of the link between mental fatigue and task disengagement," *Psychophysiology*, vol. 52, pp. 305–315, Mar. 2015.

[37] O. Blin, G. Masson, J. Azulay, J. Fondarai, and G. Serratrice, "Apomorphine-induced blinking and yawning in healthy volunteers.," *British Journal of Clinical Pharmacology*, vol. 30, no. 5, pp. 769–773, 1990.

[38] S. M. Strakowski, "Progressive behavioral response to repeated d-amphetamine challenge: Further evidence for sensitization in humans.," *Biological psychiatry*, vol. 44, 1998.

[39] S. M. Strakowski, "Enhanced response to repeated d-amphetamine challenge: Evidence for behavioral sensitization in humans.," *Biological psychiatry*, vol. 40, 1996.

[40] D. E. Redmond, "Behavioral assessment in the african green monkey after mptp administration," 2011.

[41] M. Kotani, A. Kiyoshi, T. Murai, T. Nakako, K. Matsumoto, A. Matsumoto, M. Ikejiri, Y. Ogi, and K. Ikeda, "The dopamine d1 receptor agonist skf-82958 effectively increases eye blinking count in common marmosets," *Behavioural Brain Research*, vol. 300, pp. 25–30, 2016.

[42] M. Bologna, A. Fasano, N. Modugno, G. Fabbrini, and A. Berardelli, "Effects of subthalamic nucleus deep brain stimulation and l-dopa on blinking in parkinson's disease," *Experimental Neurology*, vol. 235, no. 1, pp. 265–272, 2012.

[43] M. V. Puig, J. Rose, R. Schmidt, and N. Freund, "Dopamine modulation of learning and memory in the prefrontal cortex: Insights from studies in primates, rodents, and birds," *Frontiers in Neural Circuits*, vol. 8, 2014.

[44] A. Westbrook and T. S. Braver, "Dopamine does double duty in motivating cognitive effort," *Neuron*, vol. 89, no. 4, pp. 695–710, 2016.

[45] M. Bochove, L. Van der Haegen, W. Notebaert, and T. Verguts, "Blinking predicts enhanced cognitive control.," *Cognitive, affective & behavioral neuroscience*, vol. 13, Dec. 2012.

[46] J. R. Stroop, "Studies of interference in serial verbal reactions.," *Journal of Experimental Psychology: General*, vol. 18, pp. 643–662, 1935.

[47] T. Egner and J. Hirsch, "The neural correlates and functional integration of cognitive control in a stroop task," *NeuroImage*, vol. 24, no. 2, pp. 539–547, 2005.

[48] J. M. Bugg, "Dissociating levels of cognitive control: The case of stroop interference," *Current Directions in Psychological Science*, vol. 21, no. 5, pp. 302–309, 2012.

[49] J. Oh, M. Han, B. S. Peterson, and J. Jeong, "Spontaneous eyeblinks are correlated with responses during the stroop task," *PLoS ONE*, vol. 7, 2012.

[50] L. J. Williams, "Cognitive load and the functional field of view," *Human Factors: The Journal of Human Factors and Ergonomics Society*, vol. 24, 1982.

[51] L. J. Williams, "Tunnel vision induced by a foveal load manipulation," *Human Factors: The Journal of Human Factors and Ergonomics Society*, vol. 27, 1985.

[52] L. J. Williams, "Tunnel vision or general interference? cognitive load and attentional bias are both important.," *The American journal of psychology*, vol. 101 2, 1988.

[53] G. Underwood, D. Crundall, and P. Chapman, "Tunnel vision or general interference? cognitive load and attentional bias are both important.," *Behavioural Research in Road Safety*, 1997.

[54] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, 1943.

[55] S. J. Russell and P. Norvig, *Artificial Intelligence: a modern approach*, 3rd ed. Pearson, 2009.

[56] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[57] T. Mitchell, *Machine Learning* (McGraw-Hill International Editions). McGraw-Hill, 1997.

[58] A. F. Agarap, "Deep learning using rectified linear units (relu)," 2018.

[59] Y. LeCun, "Generalization and network design strategies," 1989.

[60] E. W. Weisstein, "Convolution," 2003.

[61] K. Holmqvist, M. Nyström, R. Andersson, R. Dewhurst, H. Jarodzka, and J. van de Weijer, "Eye tracking: A comprehensive guide to methods and measures," Jan. 2011.

[62] J. Heard, C. E. Harriott, and J. A. Adams, "A survey of workload assessment algorithms," *IEEE Transactions on Human-Machine Systems*, vol. 48, 2018.

[63] P. Gerjets, C. Walter, W. Rosenstiel, M. Bogdan, and T. O. Zander, "Cognitive state monitoring and the design of adaptive instruction in digital environments: Lessons learned from cognitive workload assessment using a passive brain-computer interface approach," *Frontiers in Neuroscience*, vol. 8, 2014.

[64] M. Hogervorst, A.-M. Brouwer, and J. Erp, "Combining and comparing eeg, peripheral physiology and eye-related measures for the assessment of mental workload," *Frontiers in neuroscience*, vol. 8, p. 322, Oct. 2014.

[65] J. L. Lobo, J. D. Ser, F. De Simone, R. Presta, S. Collina, and Z. Moravek, "Cognitive workload classification using eye-tracking and eeg data," in *Proceedings of the International Conference on Human-Computer Interaction in Aerospace*, Association for Computing Machinery, 2016.

[66] G. Wilson, C. Russell, J. Monnin, J. Estepp, and J. Christensen, "How does day-to-day variability in psychophysiological data affect classifier accuracy?" *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 54, 2010.

[67] T. Appel, P. Gerjets, S. Hoffman, K. Moeller, M. Ninaus, C. Scharinger, N. Sevcenko, F. Wortha, and E. Kasneci, "Cross-task and cross-participant classification of cognitive load in an emergency simulation game," *IEEE Transactions on Affective Computing*, pp. 1–1, 2021.

[68] K. Sharma, E. Niforatos, M. Giannakos, and V. Kostakos, "Assessing cognitive performance using physiological and facial features: Generalizing across contexts," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 4, no. 3, Sep. 2020.

[69] N. Herbig, T. Düwel, M. Helali, L. Eckhart, P. Schuck, S. Choudhury, and A. Krüger, "Investigating multi-modal measures for cognitive load detection in e-learning," in *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 88–97.

[70] A. Skulmowski and G. D. Rey, "Measuring cognitive load in embodied learning settings," *Frontiers in Psychology*, vol. 8, 2017.

[71] S. Belayachi, S. Majerus, G. Gendolla, E. Salmon, F. Peters, and M. Van der Linden, "Are the carrot and the stick the two sides of same coin? a neural examination of approach/avoidance motivation during cognitive performance," *Behavioural Brain Research*, vol. 293, pp. 217–226, 2015.

[72] A.-M. Brouwer, M. A. Hogervorst, M. Holewijn, and J. B. van Erp, "Evidence for effects of task difficulty but not learning on neurophysiological variables associated with effort," *International Journal of Psychophysiology*, vol. 93, no. 2, pp. 242–252, 2014.

[73] M. Niezgoda, A. Tarnowski, M. Kruszewski, and T. Kamiński, "Towards testing auditory–vocal interfaces and detecting distraction while driving: A comparison of eye-movement measures in the assessment of cognitive workload," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 32, Jul. 2015.

[74] H. Ayaz, M. Izzetoglu, S. Bunce, T. Heiman-Patterson, and B. Onaral, "Detecting cognitive activity related hemodynamic signal for brain computer interface using functional near infrared spectroscopy," in *2007 3rd International IEEE/EMBS Conference on Neural Engineering*, 2007, pp. 342–345.

[75] I. M., B. S., I. K., O. B., and P. A., "Functional brain imaging using near-infrared technology," vol. 26, no. 4, pp. 38–46, 2007.

[76] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller, "Deep learning for time series classification: A review," *CoRR*, 2018.

[77] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," *CoRR*, 2016.

[78] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. Zhao, "Time series classification using multi-channels deep convolutional neural networks," *WAIM 2014. LNCS*, vol. 8485, pp. 298–310, Jan. 2014.

[79] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, 2015.

[80] S. Basodi, C. Ji, H. Zhang, and Y. Pan, "Gradient amplification: An efficient way to train deep neural networks," *Big Data Mining and Analytics*, vol. 3, no. 3, pp. 196–207, 2020.

[81] S. Solhjoo, M. Haigney, E. Mcbee, J. J. G. Van Merrienboer, L. Schuwirth, A. Artino, A. Battista, T. Ratcliffe, H. Lee, and S. Durning, "Heart rate and heart rate variability correlate with clinical reasoning performance and self-reported measures of cognitive load," *Scientific Reports*, vol. 9, pp. 1–9, Oct. 2019.

[82] C. Setz, B. Arnrich, J. Schumm, R. La Marca, G. Tröster, and U. Ehlert, "Discriminating stress from cognitive load using a wearable eda device," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 2, pp. 410–417, 2010.

[83] C. Ikehara and M. Crosby, "Assessing cognitive load with physiological sensors," in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 2005, 295a–295a.

[84] H. Choi, J. J. G. Van Merrienboer, and F. Paas, "Effects of the physical environment on cognitive load and learning: Towards a new model of cognitive load," *Educational Psychology Review*, vol. 26, pp. 225–244, Jun. 2014.

Appendix A

Model Architecture Details

```
==========================================================================
Layer (type:depth-idx)                   Output Shape    Param #
==========================================================================
MLP                                      --              --
└─Sequential: 1-1                        [64, 4]         --
      ├─Flatten: 2-1                     [64, 1024]      --
      ├─Dropout: 2-2                     [64, 1024]      --
      ├─Linear: 2-3                      [64, 500]       512,500
      ├─ReLU: 2-4                        [64, 500]       --
      ├─Dropout: 2-5                     [64, 500]       --
      ├─Linear: 2-6                      [64, 500]       250,500
      ├─ReLU: 2-7                        [64, 500]       --
      ├─Dropout: 2-8                     [64, 500]       --
      ├─Linear: 2-9                      [64, 500]       250,500
      ├─ReLU: 2-10                       [64, 500]       --
      ├─Dropout: 2-11                    [64, 500]       --
      ├─Linear: 2-12                     [64, 4]         2,004
      └─Softmax: 2-13                    [64, 4]         --
==========================================================================
Total params: 1,015,504
Trainable params: 1,015,504
Non-trainable params: 0
Total mult-adds (M): 64.99
==========================================================================
Input size (MB): 0.26
Forward/backward pass size (MB): 0.77
Params size (MB): 4.06
Estimated Total Size (MB): 5.09
==========================================================================
```

Code listing A.1: Multi-Layer Perceptron (MLP)

```
==========================================================================
Layer (type:depth-idx)                   Output Shape    Param #
==========================================================================
FCN                                      --              --
└─Sequential: 1-1                        [64, 4]         --
      ├─Conv1d: 2-1                      [64, 128, 256]  4,224
      ├─BatchNorm1d: 2-2                 [64, 128, 256]  256
      ├─ReLU: 2-3                        [64, 128, 256]  --
      ├─Conv1d: 2-4                      [64, 256, 256]  164,096
      ├─BatchNorm1d: 2-5                 [64, 256, 256]  512
      ├─ReLU: 2-6                        [64, 256, 256]  --
      ├─Conv1d: 2-7                      [64, 128, 256]  98,432
      ├─BatchNorm1d: 2-8                 [64, 128, 256]  256
      ├─ReLU: 2-9                        [64, 128, 256]  --
      ├─AdaptiveAvgPool1d: 2-10          [64, 128, 1]    --
      ├─Flatten: 2-11                    [64, 128]       --
      ├─Linear: 2-12                     [64, 4]         516
      └─Softmax: 2-13                    [64, 4]         --
==========================================================================
Total params: 268,292
Trainable params: 268,292
Non-trainable params: 0
Total mult-adds (G): 4.37
==========================================================================
Input size (MB): 0.26
Forward/backward pass size (MB): 134.22
Params size (MB): 1.07
Estimated Total Size (MB): 135.56
==========================================================================
```

Code listing A.2: Fully Convolutional Network (FCN)

```
==========================================================================
Layer (type:depth-idx)                   Output Shape    Param #
==========================================================================
MCDCNN                                   --              --
└─Sequential: 1-1                        [64, 4]         --
      ├─SepConv1d: 2-1                   [64, 8, 256]    --
      │    └─Sequential: 3-1             [64, 8, 256]    64
      ├─ReLU: 2-2                        [64, 8, 256]    --
      ├─MaxPool1d: 2-3                   [64, 8, 128]    --
      ├─SepConv1d: 2-4                   [64, 8, 128]    --
      │    └─Sequential: 3-2             [64, 8, 128]    120
      ├─ReLU: 2-5                        [64, 8, 128]    --
      ├─MaxPool1d: 2-6                   [64, 8, 64]     --
      ├─Flatten: 2-7                     [64, 512]       --
      ├─Linear: 2-8                      [64, 732]       375,516
      ├─ReLU: 2-9                        [64, 732]       --
      ├─Linear: 2-10                     [64, 4]         2,932
      └─Softmax: 2-11                    [64, 4]         --
==========================================================================
Total params: 378,632
Trainable params: 378,632
Non-trainable params: 0
Total mult-adds (M): 26.25
==========================================================================
Input size (MB): 0.26
Forward/backward pass size (MB): 3.00
Params size (MB): 1.51
Estimated Total Size (MB): 4.77
==========================================================================
```

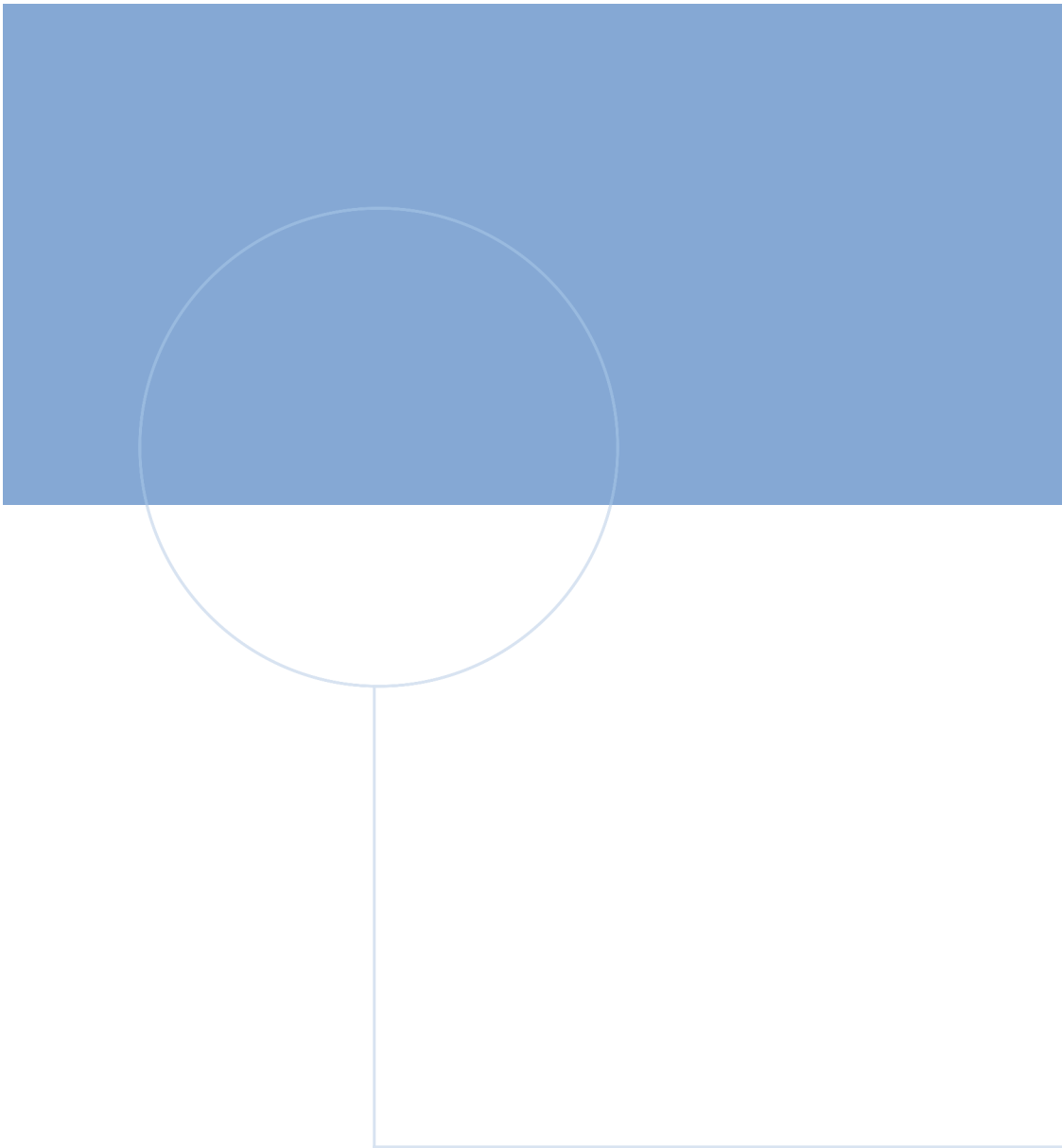
Code listing A.3: Multi-Channel Deep Convolutional Neural Network (MCDCNN)

```
==========================================================================
Layer (type:depth-idx)                        Output Shape    Param #
==========================================================================
ResNet11                                      --              --
├─Sequential: 1-1                             [64, 128, 256]  --
│    ├─ResNetBlock: 2-1                       [64, 64, 256]   --
│    │    ├─ResNet11BlockSegment: 3-1         [64, 64, 256]   35,392
│    │    ├─Sequential: 3-2                   [64, 64, 256]   448
│    │    └─ReLU: 3-3                         [64, 64, 256]   --
│    ├─ResNetBlock: 2-2                       [64, 128, 256]  --
│    │    ├─ResNet11BlockSegment: 3-4         [64, 128, 256]  197,760
│    │    ├─Sequential: 3-5                   [64, 128, 256]  8,576
│    │    └─ReLU: 3-6                         [64, 128, 256]  --
│    └─ResNetBlock: 2-3                       [64, 128, 256]  --
│         ├─ResNet11BlockSegment: 3-7         [64, 128, 256]  263,296
│         ├─Sequential: 3-8                   [64, 128, 256]  16,768
│         └─ReLU: 3-9                         [64, 128, 256]  --
└─Sequential: 1-2                             [64, 4]         --
     ├─AdaptiveAvgPool1d: 2-4                 [64, 128, 1]    --
     ├─Flatten: 2-5                           [64, 128]       --
     ├─Linear: 2-6                            [64, 4]         516
     └─Softmax: 2-7                           [64, 4]         --
==========================================================================
Total params: 522,756
Trainable params: 522,756
Non-trainable params: 0
Total mult-adds (G): 8.51
==========================================================================
Input size (MB): 0.26
Forward/backward pass size (MB): 335.55
Params size (MB): 2.09
Estimated Total Size (MB): 337.90
==========================================================================
```

Code listing A.4: 11-Layer Residual Network (ResNet11)

```
==========================================================================
Layer (type:depth-idx)                        Output Shape    Param #
==========================================================================
ResNet18                                      --              --
├─Sequential: 1-1                             [64, 512, 256]  --
│    ├─ResNetBlock: 2-1                       [64, 64, 256]   --
│    │    ├─ResNet18BlockSegment: 3-1         [64, 64, 256]   13,440
│    │    ├─Sequential: 3-2                   [64, 64, 256]   448
│    │    └─ReLU: 3-3                         [64, 64, 256]   --
│    ├─ResNetBlock: 2-2                       [64, 64, 256]   --
│    │    ├─ResNet18BlockSegment: 3-4         [64, 64, 256]   24,960
│    │    ├─Sequential: 3-5                   [64, 64, 256]   4,288
│    │    └─ReLU: 3-6                         [64, 64, 256]   --
│    ├─ResNetBlock: 2-3                       [64, 128, 256]  --
│    │    ├─ResNet18BlockSegment: 3-7         [64, 128, 256]  74,496
│    │    ├─Sequential: 3-8                   [64, 128, 256]  8,576
│    │    └─ReLU: 3-9                         [64, 128, 256]  --
│    ├─ResNetBlock: 2-4                       [64, 128, 256]  --
│    │    ├─ResNet18BlockSegment: 3-10        [64, 128, 256]  99,072
│    │    ├─Sequential: 3-11                  [64, 128, 256]  16,768
│    │    └─ReLU: 3-12                        [64, 128, 256]  --
│    ├─ResNetBlock: 2-5                       [64, 256, 256]  --
│    │    ├─ResNet18BlockSegment: 3-13        [64, 256, 256]  296,448
│    │    ├─Sequential: 3-14                  [64, 256, 256]  33,536
│    │    └─ReLU: 3-15                        [64, 256, 256]  --
│    ├─ResNetBlock: 2-6                       [64, 256, 256]  --
│    │    ├─ResNet18BlockSegment: 3-16        [64, 256, 256]  394,752
│    │    ├─Sequential: 3-17                  [64, 256, 256]  66,304
│    │    └─ReLU: 3-18                        [64, 256, 256]  --
│    ├─ResNetBlock: 2-7                       [64, 512, 256]  --
│    │    ├─ResNet18BlockSegment: 3-19        [64, 512, 256]  1,182,720
│    │    ├─Sequential: 3-20                  [64, 512, 256]  132,608
│    │    └─ReLU: 3-21                        [64, 512, 256]  --
│    └─ResNetBlock: 2-8                       [64, 512, 256]  --
│         ├─ResNet18BlockSegment: 3-22        [64, 512, 256]  1,575,936
│         ├─Sequential: 3-23                  [64, 512, 256]  263,680
│         └─ReLU: 3-24                        [64, 512, 256]  --
└─Sequential: 1-2                             [64, 4]         --
     ├─AdaptiveAvgPool1d: 2-9                 [64, 512, 1]    --
     ├─Flatten: 2-10                          [64, 512]       --
     ├─Linear: 2-11                           [64, 4]         2,052
     └─Softmax: 2-12                          [64, 4]         --
==========================================================================
Total params: 4,190,084
Trainable params: 4,190,084
Non-trainable params: 0
Total mult-adds (G): 68.43
==========================================================================
Input size (MB): 0.26
Forward/backward pass size (MB): 1509.95
Params size (MB): 16.76
Estimated Total Size (MB): 1526.97
```

Code listing A.5: 18-Layer Residual Network (ResNet18)



NTNU

Norwegian University of
Science and Technology