UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

**DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE**

CORSO DI LAUREA MAGISTRALE IN
COMPUTER ENGINEERING

# Developing Deep Learning Models for Classifying Cognitive Load States by ECG and Pupil Dilation Data Integration to Study Human Behavior


**Relatore: Prof. / Dott Loris Nanni**

**Laureando/a: Sepideh Khalatabad**

**Correlatore: Prof, /Dott Klass Bombeke**



**ANNO ACCADEMICO 2023 – 2024**

**Data di laurea 2024/04/04**

Expressions of Gratitude I extend my deepest gratitude to those who have made this thesis possible. First and foremost, my heartfelt appreciation goes to my family. To my parents for their unwavering love and support throughout my academic journey and to my brother Hossein, whose encouragement and belief in my abilities have been a constant source of strength.

I am profoundly grateful to Professor Loris Nanni at the University of Padova for his guidance and invaluable insights, which have significantly enriched my research experience. His mentorship has been instrumental in my academic development.

Special thanks are due to my esteemed supervisor, Professor Klaas Bombeke, whose expertise and attention to detail have greatly enhanced the quality of my work. His patience and willingness to educate me have been genuinely inspiring even when I took missteps. I am also indebted to Dr. Aleksandra Zheleva and Dr. Jonas De Bruyne at Imec-mict-UGent in Gent University for their constructive feedback, unwavering support, and wise counsel. Their commitment to excellence and innovation has profoundly influenced my growth as a researcher.

I also express my deep appreciation for my friend Yeganeh, whose support, camaraderie, and unwavering faith in me have been of immense comfort during this challenging but rewarding journey.

To all, I express my sincere appreciation and gratitude for accompanying me on this scholarly voyage. Your support has been a beacon of light in moments of doubt and has consistently guided me toward realizing my academic goals.

**Abstract**

This thesis investigates the classification of cognitive load states using biometric sensor data, specifically within the context of N-back tasks performed in a Virtual Reality (VR) environment. Cognitive load is widely studied, especially in contexts like cognitive ergonomics, training and education, and UX research. While subjective methods exist to measure cognitive load, researchers are looking into ways to objectively and, in particular, automatically classify different cognitive load states. Such automatic detection would benefit various contexts, ranging from real-time operator monitoring to adaptive training ( flow). In this thesis project, we aim to delve into the development of automated methods for classifying cognitive load states. Our goal is to create a robust framework that can objectively assess and differentiate between varying levels of cognitive load without relying on subjective human assessments. The project mainly involves the exploration of various data sources, sensor technologies, and machine learning techniques to develop a reliable and practical system for automatic cognitive load classification. This project's Successful completion will contribute to advancing human-centered technology and research and provide valuable insights into optimizing cognitive ergonomics across a wide spectrum of applications—think adaptive training systems, improved worker productivity and wellbeing, UX research, and more.

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# List of Code Snippets

# List of Acronyms

**VR** Virtual Reality

**UX** User Experience

**VE** Virtual Environment

**ECG** Electrocardiogram

**CLT** Cognitive Load Theory

**ML** Machine Learning

**DL** Deep Learning

**MLP** Multi-layer Perceptron

**GRU** Gated recurrent unit

**ANNs** artificial neural networks

**BN** Batch Normalization

**RNN** Recurrent neural network

**TSP** Time Series Prediction

**EDA** Exploratory Data Analysis

**SCV** Stratified K-Fold cross-validatio

**SVM** Support Vector Machine

**AR** Augmented Reality

**AI** Artificial Intelligence

**TSP** Time Series Prediction

**PVC** Premature Ventricular Contraction

**APC** Atrial Premature Contraction

**LBBB** Left Bundle Branch Block

**RBBB** Right Bundle Branch Block

# 1 Introduction

The advent of extended reality (XR) technologies marks a transformative era in training and learning scenarios, where biofeedback measures like heart rate, eye gaze, and pupil dilation play a pivotal role. These measures are essential for tailoring adaptive and practical XR training experiences that meet the growing demands for personalized learning paths. Despite the advancements in XR technologies equipped with physiological sensors, a fundamental challenge persists: accurately interpreting these biofeedback signals in light of inter-individual physiological variances. This thesis underscores the significance of utilizing biometrics to measure how an individual's physiological signals fluctuate in response to cognitive state changes, an endeavor critical for developing precise XR assessments and adaptive training programs [145].

In Virtual Reality (VR), the classification and assessment of cognitive load have emerged as critical factors in enhancing user experiences within technology-driven environments. However, relying on subjective methodologies has often failed to capture the essence of cognitive load in VR's dynamic and immersive settings. This research identifies a gap in the current literature – integrating multi-modal biometric data, including electrocardiogram (ECG) and pupil dilation, is crucial for a holistic understanding of cognitive load within these environments.

To address the gap identified in current research, this thesis proposes an innovative methodology by constructing various frameworks to classify cognitive loads using both augmented and real datasets. By developing a robust deep-learning model, this study aims to transform the assessment and real-time

adaptation of cognitive load significantly. This model's capability to seamlessly integrate and analyze a wide range of biometric data is crucial for tasks based on Virtual Reality (VR), necessitating a detailed and objective grasp of cognitive load.

The cornerstone of this thesis is creating and validating a comprehensive deep learning-based framework. This framework is specifically designed to classify cognitive load states, synergizing ECG and pupil dilation data within N-back tasks in a VR environment. Such a methodological approach is expected to enhance the precision of cognitive load assessments significantly. Moreover, by offering a scalable and efficient solution for real-time cognitive load monitoring across various VR applications, this research contributes profoundly to advancing human-centered technology, promising to reshape the landscape of cognitive ergonomics, education, and user experience design in immersive environments.

## 1.1 PROBLEM STATEMENT

The emergence of extended reality (XR) technologies for training and immersive experiences has underscored a significant challenge: the need for precise interpretation and integration of biometric data to assess cognitive load. Despite the progress in XR technology, including headsets with physiological sensors, the reliance on subjective measures for cognitive load evaluation in Virtual Reality (VR) environments reveals a gap. This gap is particularly evident in needing a holistic approach to leveraging multimodal biometric data, such as electrocardiogram (ECG) and pupil dilation, for a dynamic and objective analysis.

Acknowledging this shortfall, this thesis addresses the limitations of current deep learning models in providing a real-time and accurate assessment of cognitive load in VR settings. Traditional methods of cognitive load assessment often overlook the subtleties of individual physiological responses, failing to capture the complexity of cognitive states. Moreover, despite the application of deep learning models in parsing physiological data, there exists a need for models adept at efficiently utilizing and analyzing a diverse array of biometric information for immediate cognitive load evaluation amidst the varied tasks presented in VR environments. In response to these challenges, this research endeavors to construct an extensive deep-learning framework to integrate and interpret ECG and pupil dilation data cohesively. This innovative framework seeks to re-

fine cognitive load classification's precision, offering a scalable and flexible solution for instantaneous monitoring across a spectrum of VR applications. By addressing these critical gaps, the study aims to propel XR technologies forward, enhancing their utility in training, education, and ergonomic design, thereby making a substantial contribution to the advancement of XR applications.

## 1.2 OBJECTIVES OF STUDY

The innovation in this research lies in the novel use of deep learning models for interpreting and integrating biometric signals in real-time. Deep learning, known for its ability to handle large, complex datasets, presents a unique opportunity to glean insights from the nuanced physiological responses captured during VR interactions. To this end, two distinct models will be compared: the first leverages traditional deep learning architectures, and the second incorporates a novel augmentation technique tailored for psychological datasets. This comparative analysis aims to evaluate the effectiveness of each model in classifying cognitive load and identify the potential advantages of the new augmentation technique in enhancing model performance.

Furthermore, the augmentation technique being introduced is groundbreaking in its approach to handling psychological data. Traditional methods often need help with the variability and subtlety of biometric signals, especially in a field as complex as cognitive load assessment. By incorporating this new technique, the research hopes to address these challenges, offering a more nuanced and accurate analysis of cognitive load states. This is particularly crucial in VR environments, where user experiences are highly dynamic and interactive, demanding a more sophisticated approach to data interpretation.

## 1.3 APPLICATIONS

The application's initiation ushers in the presentation of an intuitively designed graphical user interface (GUI), which serves as the central platform for the trainer's interaction with the system. This interface has been carefully designed to ensure smooth entry of user-specific details, precise control over data channels—including their enhancement or removal—and the adjustment of key parameters relevant to training sessions.

The construction of the GUI harnesses the capabilities of PySimpleGUI, a Python toolkit celebrated for its streamlined approach to GUI construction, allowing for rapid development without sacrificing functionality. In tandem, the application employs pylsl [114], an esteemed library for interfacing with the Lab Streaming Layer (LSL), thereby enabling robust data acquisition and streaming processes. The integration of peripheral Bluetooth devices is elegantly handled via the implementation of bleak [15], a specialized Python library that provides comprehensive management of Bluetooth operations, ensuring stable connectivity and communication within the application's ecosystem.

## 1.4 LIMITATIONS

This Master's thesis delves into classifying cognitive load states within a Virtual Reality (VR) context using biometric sensor data. Throughout this investigation, several notable limitations have surfaced. The most prominent challenge is the relatively small dataset, consisting of data from merely 21 participants. This limitation, primarily due to the labor-intensive nature of VR testing, significantly narrows the diversity and generalizability of the findings. Moreover, the constrained dataset size presents hurdles for machine learning algorithms and sophisticated learning approaches, heightening the risk of model overfitting and compromising the models' robustness. The project also faces constraints in computational resources, which in turn restricts the complexity of the neural network models that could be designed and trained. Additionally, the necessity to focus on sensor-level data, dictated by these limitations, may inadvertently narrow the research scope, possibly omitting deeper insights achievable through a broader analysis encompassing higher-level data interpretations or integrations. Despite these obstacles, this project endeavors to maximize the utility of the available data and resources, aiming to garner significant insights within the set limitations.

# 2 Background

This chapter outlines the essential theory behind the work in this Master's thesis, focusing on an in-depth exploration of the main concepts examined.

## 2.1 VIRTUAL REALITY

*Virtual reality* (VR) is a concept with various interpretations, commonly understood as the integration of virtual objects in a virtual setting [69]. It can also be broadly defined as a computer-generated representation of real-world scenarios, offering immersive experiences through mainly auditory and visual stimuli [62]. VR creates an environment that simulates three dimensions, making users feel present. According to Desai, Ajmera, and Mehta (2014) [31], this digital replication of real life enables active modification, engagement, and observation of the environment. Brooks (1999) [16] expanded on this, discussing the range of gestures and actions available to users in VR. High-quality virtual environments in advanced VR systems enhance user interaction and navigation, simulating real-life experiences closely [69]. Ivan Sutherland [94], a key figure in computer graphics and VR, envisioned a space where virtual objects like balls would have realistic weight and texture, allowing for actions like throwing or bouncing. This vision also included the potential for more intricate objects, like handcuffs, to be realistically emulated, showcasing VR's capacity to replicate the tactile sensation of physical objects with remarkable precision [94].

To be more precise, virtual reality falls into two main categories: immersive and non-immersive [133]. Immersive VR, typically accessed via a head-mounted

display (HMD), is distinguished by its ability to substitute a user's real-world sensory inputs with those from a virtual environment [2]. This form of VR is characterized by deep immersion, high interactivity, and intense user engagement, drawing the user entirely into the virtual realm and away from reality [10].

Head-mounted displays (HMDs) are instrumental in enhancing this immersive experience. They allow users to move through VR environments following their head and eye movements, effectively obscuring the real world. This feature goes beyond mere exploration and includes interaction with virtual entities and objects, as discussed by Sanchez-Vives& Slater (2005) [104].

On the other hand, non-immersive VR offers a virtual experience without completely disconnecting users from their physical surroundings. This type allows for interactive but less total engagement [10]. Understanding this distinction is essential for grasping current technology's range of virtual experiences [10].

The HTC VIVE Pro Eye [1] serves as our chosen Virtual Reality (VR) device [135]. This advanced headset is integral to our work, providing high-resolution displays with 1440x1600 resolution per eye, which is essential for the detailed visual requirements of our project. The eye-tracking technology in select models of the VIVE Pro Eye is particularly beneficial, offering precise tracking capabilities crucial for calibration accuracy. Its ergonomic design, balanced form, lighter weight, and adjustable sizing ensure comfort during prolonged use sessions. The integrated audio and noise-cancellation microphone also contribute to an immersive experience, enhancing the device's overall effectiveness in our calibration processes [39]. The HTC VIVE Pro Eye's capabilities align seamlessly with the requirements of our project, making it an invaluable tool in our VR-based calibration endeavors[134].

[1]https://www.lib.ncsu.edu/devices/vive-pro-eye


Figure 2.1: HTC VIVE Pro Eye.

## 2.2 COGNITIVE LOAD

Cognitive Load (CLT) theory is rooted in understanding cognitive architecture. It highlights the contrast between the working memory, which has a restricted capacity and a limited duration for processing incoming information [93, 58], and the long-term memory, known for its nearly limitless capacity [122]. This distinction is crucial for comprehending how memory functions in the assimilation and understanding of content, with adjustments for the distinct cognitive structures of individuals [122, 81].

In working memory, an individual manages around $7 \pm 2$ pieces of information simultaneously. This capacity diminishes when the task involves not just recall but active processing, especially when these pieces of information are interconnected and require synthesis [28]. Moreover, the inherent limitations of working memory pose considerable challenges in learning scenarios, particularly with complex tasks. For instance, mastering a new language's grammar, which entails integrating various elements like subject, predicate, and object, demands significantly more from working memory than straightforward tasks, such as vocabulary memorization [122]. However, as knowledge is integrated into long-term memory and organized into cognitive schemas, the burden on working memory decreases. In this scenario, a schema is perceived as a single informational unit within working memory, streamlining the processing effort [122]. This phenomenon underscores how prior expertise or familiarity with a task can substantially alleviate the cognitive load. Furthermore, with the repet-

itive performance of tasks or their components, cognitive schemas become automated, obviating the need for controlled processing and thereby liberating working memory capacity [107].

Cognitive load, the total amount of information that working memory can handle at any given moment, encompasses three distinct types: intrinsic, about the task's inherent complexity; extraneous, referring to environmental factors that exacerbate cognitive demands; and germane, which involves connecting new information to existing knowledge stored in long-term memory [120, 8]. The intrinsic cognitive load is influenced by the learner's skill level relative to the complexity of the subject matter. According to Paas et al., this load is significantly affected by the level of element interactivity within the learning materials. Materials rich in interactive elements are more challenging as they necessitate concurrently processing multiple components. Conversely, materials with lower interactivity facilitate independent learning due to reduced simultaneous processing demands [89]. The germane load is linked to the cognitive effort required to manage intrinsic load, promoting integrating or adapting new concepts while providing suitable challenges to learners [121]. For example, the germane load is engaged in language acquisition when learners actively compare their native language with a new language, fostering deeper cognitive engagement [66].On the other hand, extraneous load arises from suboptimal instructional designs that inadvertently increase cognitive strain, impeding learning. This can occur when instructional methods inadvertently amplify cognitive load, such as employing a diagram without a clear correlation with its textual examples, leading to divided attention [22].

CLT emphasizes the critical role of working memory in the efficient delivery of information [120]. Strategies for optimizing working memory utilization include varying the presentation of information or employing dual modalities, such as visual-spatial and phonological processing, to manage the complexity of learning materials effectively [122]. This approach balances intrinsic, germane, and extraneous loads to maximize learning outcomes. Long-term memory acts as a knowledge repository, facilitating information transfer to working memory, which is crucial for accessing and applying stored knowledge. Constructing individual schemas involves transforming disparate pieces of information into a coherent whole and retrieving this consolidated knowledge from long-term memory. This process depends on the individual's working memory capacity [122]. Therefore, a fundamental principle of CLT is the individual's cognitive

capacity to process information, which ultimately determines the efficacy and efficiency of their learning experiences [122, 81].

### 2.2.1 MEASURING COGNITIVE LOAD BY ECG

*An Electrocardiogram* (ECG) is a tool used to measure heart activity by recording the heart's electrical activity over time [34]. It is achieved by placing electrodes on the skin. ECG has been used to measure cognitive load by analyzing the changes in heart activity that occur in response to cognitive tasks. Several studies have explored using ECG for cognitive load assessment [20]. Evaluation of cognitive load is a burgeoning area in human-machine interaction (HMI), particularly in mixed-initiative systems where human and automated systems collaborate. The study by Wilson and Scannella [20] underscores the potential of using ECG-based metrics to monitor cognitive states in real-time, a crucial aspect for the dynamic allocation of decision-making authority in these systems. Their groundbreaking work employed a multi-level letter-span task to elicit varying cognitive loads from participants. This approach was pivotal in assessing the efficacy of online heart rate (HR) and heart rate variability (HRV) metrics compared to traditional offline methods. The methodology's strength lies in its ability to simulate real-world tasks where cognitive load varies, providing a robust framework for ECG data analysis [53] research highlights the utility of ECG (electrocardiogram) and other physiological measures in assessing cognitive load. The study's findings demonstrate that ECG metrics, specifically the median absolute deviation of the electrocardiogram, coupled with median heat flux measurements, were markedly effective in distinguishing between low and high levels of cognitive load provided a comprehensive analysis of the use of electrocardiogram (ECG) data in the assessment of cognitive load [143]. Their research highlighted the nuanced capabilities of ECG metrics to detect subtle changes in cognitive stress levels, particularly in high-demand multitasking environments. Yoo, Kim, and Hong (2023) [143] emphasized the importance of ECG as a non-invasive and effective tool for real-time monitoring of cognitive load, thereby offering significant insights into optimizing learning and work environments. Their work underscores the potential of ECG data in enhancing adaptive learning systems by dynamically adjusting to the cognitive states of individuals [20, 43, 5]. Recent advancements have enabled the real-time analysis of ECG to monitor cognitive load, particularly in mixed-initiative systems, where

humans and machines collaborate and share decision-making responsibilities. Integrating online ECG-based features in such systems can enhance the adaptability and safety of the human-machine interaction by dynamically responding to the operator's cognitive state.

## 2.3 ECG SIGNAL

An Electrocardiogram (ECG) signal, a pivotal diagnostic tool in cardiology, is characterized by its distinct waveforms: the P, Q, R, S, and T waves. Each wave reflects a specific phase in the cardiac cycle, corresponding to the heart's electrical activity as it propagates through various cardiac structures [51]. These waveforms' morphology, orientation, and frequency provide critical insights into cardiac health, revealing conditions such as arrhythmias and ischemic heart disease.

The core component of an ECG signal, the QRS complex, is particularly crucial. This complex is marked by its high amplitude, representing ventricular depolarization, making it the most conspicuous and algorithmically detectable feature of the ECG signal. The precision in identifying the QRS complex, which encompasses the heart's electrical activity through successive positive and negative peaks, is vital for accurately determining the heart rate (HR). Heart rate calculation relies on measuring the Inter-Beat Interval (IBI)—the duration between two consecutive heartbeats, also known as the beat-to-beat or RR interval [75, 20].

Furthermore, the R wave within the QRS complex signifies the transmission of electrical impulses across the central segment of the ventricular walls. The accurate detection of the R-Peak is essential for calculating the IBI, which is instrumental in ECG signal analysis. The heart rate, expressed in beats per minute, is determined by counting the R peaks over a specified period. The R-R interval, extending from the peak of one R wave to the next, represents the temporal gap between two successive QRS complexes, serving as a key metric in evaluating heart rhythm and the function [75, 20].

The comprehensive analysis of an ECG extends beyond the QRS complex to include other significant components, such as the P wave, which denotes atrial contraction, and the P–R segment and interval. The P–R segment signifies a pause allowing for ventricular filling, attributed to an AV node delay, while the P–R interval reflects the onset period before ventricular depolarization begins.

Variations in the P–R interval indicate the parasympathetic nervous system's influence on the heart, providing further insights into cardiac autonomic regulation [36].

Figure 4.1 captures these key ECG components, illustrating the intricate process of cardiac electrical activity and its significance in diagnosing and understanding cardiac conditions.[36].

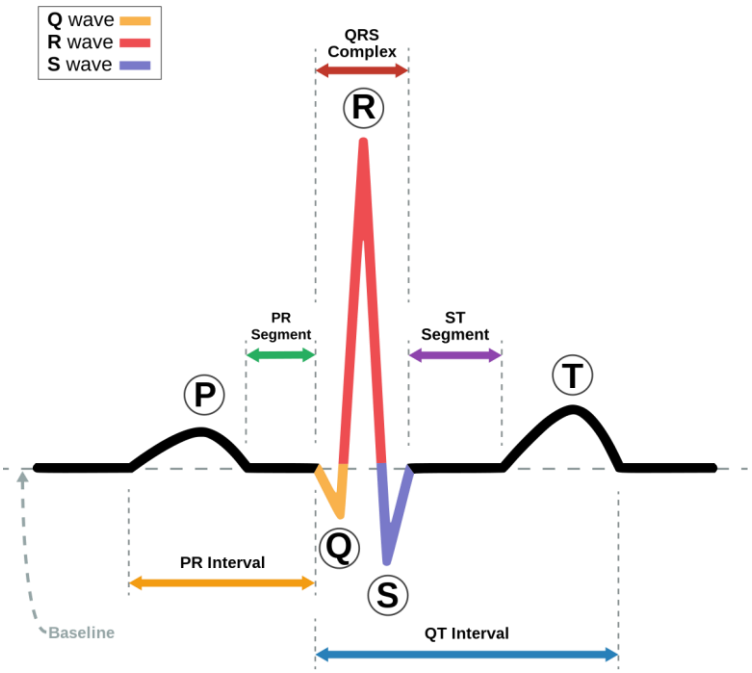


Figure 2.2: Electrocardiography

## 2.4 EYE-TRACKING SENSOR

Eye-tracking technology has become a cornerstone in understanding cognitive processes during human-computer interaction. By employing near-infrared light, this sensor-based technology accurately tracks and records the movements of the eyes, offering real-time insights into a person's gaze direction. Since its advent in the 1980s, eye tracking has played a pivotal role in exploring how individuals interact with computer interfaces, significantly influencing the design and improvement of these interfaces [138] as suggested by Hoffman, the foundational principle that visual attention guides eye movement—where the gaze

is directed by where the attention shifts—is central to the functionality of eye-tracking systems. These systems employ infrared cameras to capture detailed reflections from the cornea and pupil as users engage with visual stimuli on a screen, such as images, colors, or text, thus enabling the precise estimation of gaze direction with exceptional spatial and temporal accuracy [50]. Eye tracking's capabilities extend beyond mere observation of eye positions and movements; it provides a comprehensive analysis of visual attention, tracking and analyzing the focus of gaze across various objects and the duration of such engagements [41].

The applications of eye-tracking technology are manifold, encompassing:

- **Monitoring blinking behaviors:** This aspect of eye tracking offers insights into user focus and cognitive load, as blinking patterns can reflect engagement and fatigue levels.
- **Detecting Overlooked Information:** By identifying elements that users miss, researchers can enhance the design of interfaces to ensure critical information is more noticeable.
- **Evaluating Pupil Responses:** Analyzing how pupils react to different visual or emotional stimuli provides a deeper understanding of user responses to content, aiding in the creation of more engaging and effective designs.
- **Enhancing Human-Computer Interaction:** The insights gained from eye-tracking research are invaluable in refining the usability of computer interfaces. They ultimately contribute to the advancement of machine learning and improve the synergy between humans and computers.

## PUPIL DIAMETER

The human pupil exhibits dynamic changes in size in response to various stimuli, which can be broadly categorized into three types: the Pupil Light Response (PLR), the Pupil Near Response (PNR), and the Psychosensory Pupil Response (PPR). These responses signify the pupil's contraction in bright light or during close vision tasks (PLR and PNR, respectively) and its dilation in response to cognitive demands, such as increased mental effort or arousal (PPR) [7].

Pupillary reactions display a dual nature: they are reflexive, showing consistent qualitative responses to specific stimuli (e.g., light exposure always results in pupil constriction), and also exhibit voluntary aspects, influenced by cognitive

processes (e.g., focusing on a light source leads to more pronounced constriction) [14, 13]. This complexity of pupil responses, encompassing involuntary reflexes and voluntary control, parallels other ocular movements such as saccades and smooth pursuit, highlighting the intricate interplay between sensory stimuli and cognitive functions [148].

## 2.5 IMPORTANCE OF PUPIL DILATION DATA IN COGNITIVE LOADS

Pupil dilation is a pivotal marker for cognitive load, providing invaluable insights into an individual's mental workload during task execution [67]. The correlation between increased task complexity and pupil diameter expansion is well-documented, underscoring the utility of pupillary changes as a gauge for cognitive load [95].

Key aspects of pupil dilation data, critical for assessing cognitive loads, include:

1. **Cognitive Load Indicator:** Pupil dilation reacts sensitively to changes in visual field brightness and the cognitive demands of visual tasks, serving as a direct indicator of cognitive load [95].

2. **Task-Induced Pupillary Changes:** The association between pupil size and cognitive load has been examined across various contexts, such as driving and educational settings, demonstrating how pupillary changes can reflect cognitive load [79].

3. **Assessing Task Difficulty:** Data on pupil response can help determine the cognitive load in diverse settings, such as educational video games. Comparisons with baseline measurements suggest the effectiveness of pupil dilation as an indicator of cognitive load [79].

4. **Comparative Analysis with Other Metrics:** Pupil dilation stands out as a practical metric for cognitive load assessment when juxtaposed with other measures such as electroencephalography (EEG), skin conductance, and cardiac metrics, showcasing its applicability in real-world scenarios [97].

5. **Underpinning Cognitive Load Theory:** Reflecting the principles of Cognitive Load Theory, which posits that cognitive processing capacities influence learning efficiency, pupil dilation has been extensively employed to investigate cognitive load in various educational contexts [46].

## 2.6 MACHINE LEARNING

Machine learning is a pivotal technology in data science and artificial intelligence, tasked with developing computer systems capable of autonomously acquiring new capabilities and improving performance. This advancement is achieved through data analysis and experience accumulation, aiming to enhance a performance measure via training experience in executing a specified task [61]. Machine learning methodologies are diverse, encompassing supervised, unsupervised, semi-supervised, active, transfer, multi-task, and reinforcement learning, each tailored to specific applications [61]. Supervised learning, identified as the predominant approach, focuses on constructing models designed to predict outputs from given input data, wherein labels for all or most data points are pre-defined [61].

In supervised learning, tasks typically include classification and regression, employing techniques such as k-nearest neighbors, support vector machines, and linear discriminative analysis. A burgeoning field of application for supervised machine learning is the classification of ECG signals. This area has witnessed a surge in research efforts, introducing many algorithms and methodologies designed to enhance the accuracy and efficiency of ECG signal analysis [9].

The process of classifying ECG signals with supervised machine learning encompasses two critical steps: feature extraction and classifier model selection. Initially, diverse features are derived from each dataset, which is then utilized by various classifiers, such as Support Vector Machine (SVM) and Multilayer Perceptron (MLP), to categorize ECG signals into distinct categories including standard, Premature Ventricular Contraction (PVC), Atrial Premature Contraction (APC), Left Bundle Branch Block (LBBB), Right Bundle Branch Block (RBBB), and Paced (PACE) heartbeats [9].

Further expanding the application of machine learning and artificial intelligence, recent studies have focused on detecting cardiac rhythm disorders through ECG signals. These studies emphasize the significance of feature extraction and selection, employing various preprocessing techniques, features, and classifiers to achieve this goal [55]. This evolution underscores the continuous innovation in machine learning techniques and their critical role in advancing medical diagnostics.

## 2.7 DEEP LEARNING: A PINNACLE OF MACHINE LEARNING

Deep learning (DL) represents a sophisticated advancement in machine learning (ML), distinguished by employing neural networks comprising at least three layers. These multi-layered networks are inspired by the human brain's structure and functionality, albeit far from matching its complexity. They are engineered to process and learn from extensive datasets, leveraging the depth of their architecture to enhance learning capabilities. While a single-layer neural network can perform basic predictions, introducing additional hidden layers significantly increases the model's precision and ability to make more nuanced predictions [72, 136]. ML and DL are integral components of the artificial intelligence (AI) spectrum, contributing to its diverse capabilities [24].

The application of deep learning models in classifying ECG signals has been a focal point of recent research, demonstrating promising results that could revolutionize diagnosing cardiac anomalies and advancing intelligent healthcare systems [110]. A notable study by Ribeiro et al emphasized the superiority of deep neural networks (DNNs) in automatically classifying S12L-ECG signals, showcasing their advantage over commercial ECG software that predominantly relies on traditional signal processing techniques [101]. This progress underscores deep learning's potential in enhancing healthcare diagnostics, offering a more accurate and efficient means of interpreting ECG signals. The ability of DNNs to outperform existing methodologies highlights the advancements in AI and paves the way for the development of more sophisticated and reliable diagnostic tools.

## 2.8 TIME SERIES CLASSIFICATION WITH ARTIFICIAL NEURAL NETWORKS

Artificial Neural Networks (ANNs) have been recognized for their exceptional ability to discern patterns and subtleties within complex datasets. This feature is attributed to their capacity to handle high-dimensional data and their inherently differentiable nature [76]. These qualities contribute significantly to the versatility and power of ANNs, setting the stage for their application across various domains, including time series classification.

# ANNS IN TIME SERIES CLASSIFICATION

The field of time series classification has benefitted immensely from the development and application of specific ANN models designed to address the unique challenges presented by time-dependent data. These models include: (Gated Recurrent Units), S4 models and Transformers, as cited in [44, 27] and [132] Notably, GRU stand out for their unique features and functionalities that render them particularly adept at managing and classifying sequentially ordered data, a central theme of my thesis.

## 2.8.1 BASICS OF THE GRU-DROPOUT ARCHITECTURE

The development of the Gated Recurrent Unit (GRU) by Cho et al. [25] in 2014 marked a significant advancement in recurrent neural networks (RNNs). GRUs were introduced to tackle the notorious vanishing gradient problem that plagues standard RNNs, hindering their ability to process long sequences effectively. Unlike traditional RNNs, GRUs incorporate update and reset gates, which enable the network to manage information flow better, making them adept at retaining relevant information over extended sequences.

Cho et al.'s work [25] lays the foundational framework for GRUs, distinguishing them as a variant of the Long Short-Term Memory (LSTM) networks. Both GRUs and LSTMs are designed to remember information for long periods. Still, GRUs simplify the model architecture by combining the input and forget gates into a single update gate, thus reducing the complexity and computational requirements [131]. This design choice is pivotal in making GRUs efficient and powerful for tasks involving sequential data, such as natural language processing and time series analysis.

The architecture's efficiency stems from its two key components: the update and reset gates. These gates regulate the flow of information, deciding what to retain and discard, thereby addressing the vanishing gradient issue more effectively than their predecessors [65, 27]. The update gate, in particular, is critical in determining how much past information needs to be passed along to the future, allowing the GRU to capture dependencies across longer time spans.

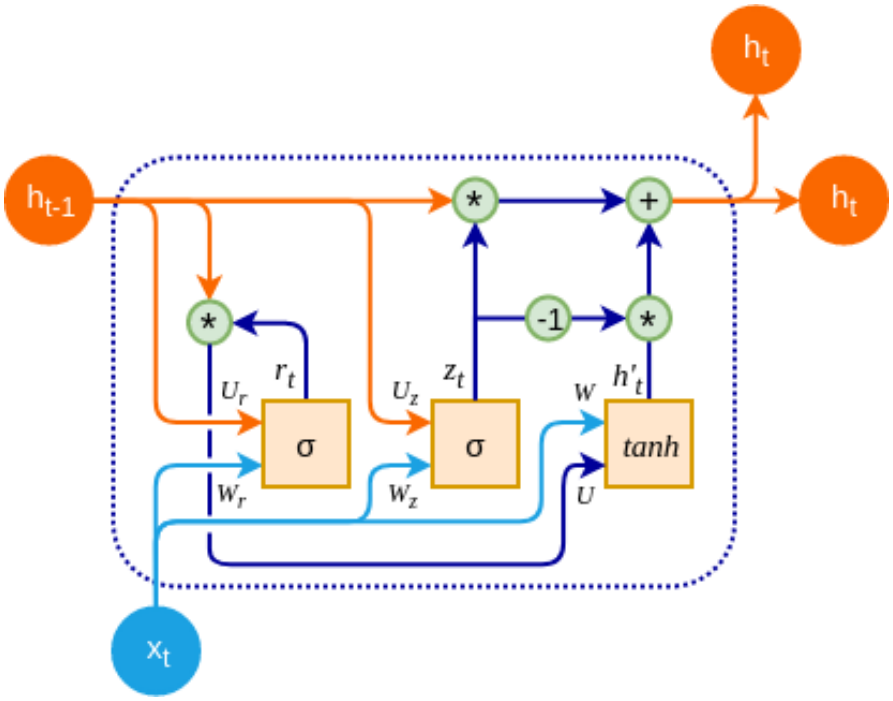


Figure 2.3: The GRU-Dropout architecture.

## GATED RECURRENT UNIT (GRU) FORMULAS

The operational core of the GRU is encapsulated in its formulas, which mathematically represent the functions of the update and reset gates, as well as the computation of the candidate and final hidden states. These formulas are:

- **Update Gate** ($z_t$):

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \tag{2.1}$$

  where $\sigma$ is the sigmoid function, $W_z$ is the weight matrix for the update gate, $h_{t\_1}$ is the previous hidden state, $x_t$ is the input at time step $t$, and $b_z$ is the bias for the update gate[1, 49].

- **Reset Gate** ($r_t$):

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{2.2}$$

  Here, $W_r$ is the weight matrix, and $b_r$ is the bias for the reset gate. The reset gate determines how much of the past information to forget, which is crucial for the model to focus on the most relevant information for the task at hand [49].

- **Candidate Hidden State** ($\tilde{h}_t$):

$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t] + b) \tag{2.3}$$

  where $W$ and $b$ are the weight matrix and bias for the candidate hidden state, and $\tanh$ represents the hyperbolic tangent function.

- **Final Hidden State** ($h_t$):

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{2.4}$$

This equation determines the final hidden state, blending the previous state with the new candidate hidden state, controlled by the update gate.

### 2.8.2 MULTI-LAYER PERCEPTRON (MLP)

*The Multi-Layer Perceptron* (MLP), a key figure in the landscape of supervised neural networks, is distinguished by its three-layer architecture: the input layer, one or more hidden layers, and the output layer. Each layer comprises neurons, also called nonlinear computational elements, facilitating a dynamic flow of information from the input to the output layer through the intermediary hidden layers. This sophisticated structure has made the MLP an optimal choice for many approximation tasks across static and dynamic settings [11].

A hallmark of the MLP is its intricate network of neurons spread across multiple layers. The inclusion of hidden layers introduces depth and complexity, enabling the network to model intricate patterns in data [11]. Another critical aspect of MLPs is the flexibility in choosing activation functions. Unlike basic perceptrons constrained to functions like ReLU or sigmoid, MLP neurons can utilize a broader range of activation functions, enhancing their ability to capture non-linear relationships [11]. This versatility renders MLPs highly effective for diverse applications, from image recognition to natural language processing.

MLPs employ a structured learning process that begins with data input through the input layer, followed by forward propagation through the hidden and output layers. This process is vital for adjusting the network based on the input data. The network then evaluates its performance by comparing predicted results against actual outcomes, aiming to minimize discrepancies or errors. Backpropagation plays a crucial role in this optimization process, allowing for the adjustment of weights by calculating the error gradient with respect to each weight. This mechanism ensures continuous improvement and accuracy of the model [98].

#### FORWARD PROPAGATION

Forward propagation essentially involves taking an input feature vector as

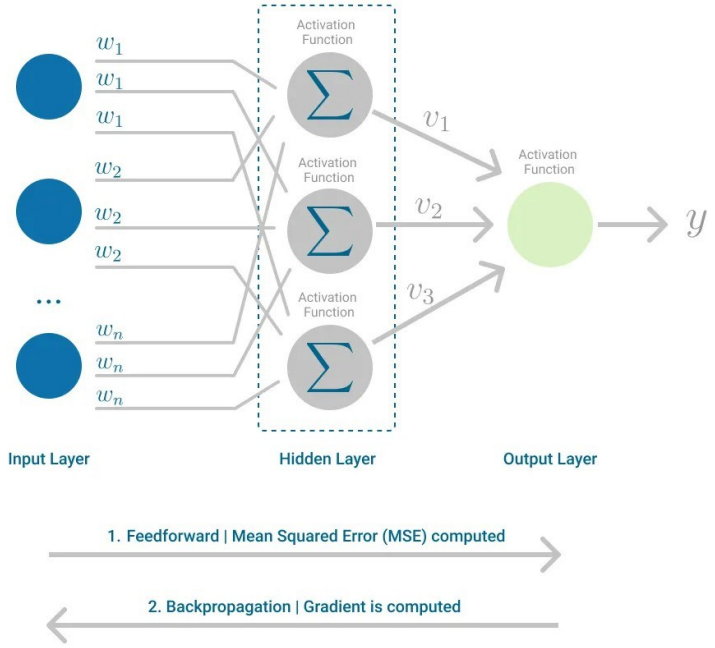


Figure 2.4: Multilayer Perceptron, emphasizing the Forward Feeding and Backward Error Propagation phases.

$X = (x_0, x_1, \ldots, x_{n_0})$ , the weight matrix for layer $l$ as $W^{[l]}$, the bias vector as $b^{[l]}$, and the activation function as $g^{[l]}$. Then, for layer $l$, the forward propagation to the next layer can be expressed as[14]:

$$Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]} \tag{2.5}$$

$$A^{[l]} = g^{[l]}(Z^{[l]}) \tag{2.6}$$

Here, $A^{[0]} = X$ is the input to the first layer, $Z^{[l]}$ is the linear combination of weights and inputs, and $A^{[l]}$ is the output of layer $l$ after applying the activation function.

For the output layer $L$, the predicted output $\hat{Y}$ is:

$$\hat{Y} = A^{[L]} = g^{[L]}(Z^{[L]}) \tag{2.7}$$

where $Z^{[L]} = W^{[L]}A^{[L-1]} + b^{[L]}$.

These equations collectively represent the forward propagation process in a neural network.

## BACK-PROPAGATION

Assuming we have a loss function denoted as $L$, the gradients of this loss with respect to the weights $W^{[l]}$ and biases $b^{[l]}$ in layer $l$ can be computed using the chain rule. For each neuron, the error gradient is propagated backward from the output layer to the input layer. The updated rules for the weights and biases can be expressed as follows:

$$\frac{\partial L}{\partial W^{[l]}} = \frac{\partial L}{\partial A^{[l]}} \cdot \frac{\partial A^{[l]}}{\partial Z^{[l]}} \cdot \frac{\partial Z^{[l]}}{\partial W^{[l]}} \tag{2.8}$$

$$\frac{\partial L}{\partial b^{[l]}} = \frac{\partial L}{\partial A^{[l]}} \cdot \frac{\partial A^{[l]}}{\partial Z^{[l]}} \tag{2.9}$$

Here, $\frac{\partial A^{[l]}}{\partial Z^{[l]}}$ is the derivative of the activation function at layer $l$, and $\frac{\partial L}{\partial A^{[l]}}$ is the derivative of the loss with respect to the activation of layer $l$.

The weights are then updated by subtracting a fraction of the gradient, determined by the learning rate $\eta$:

$$W^{[l]} = W^{[l]} - \eta \cdot \frac{\partial L}{\partial W^{[l]}} \tag{2.10}$$

$$b^{[l]} = b^{[l]} - \eta \cdot \frac{\partial L}{\partial b^{[l]}} \tag{2.11}$$

The process is repeated iteratively for each network layer moving from the output layer back to the first hidden layer.

## 2.9 EVALUATION METRICS

In evaluating machine learning (ML) algorithms, we took a comprehensive approach by utilizing a set of key performance metrics. These metrics encompass Accuracy, Error Rate, Recall (also known as Sensitivity), Specificity, Precision, and the F1-Score. Our methodology aligns with established practices outlined in prior research studies [38, 137].

The evaluation process commenced with the computation of Sensitivity (or Recall), Specificity, Precision, and Accuracy. Sensitivity, also referred to as Recall, quantifies the correct identification of actual positive cases, which is vital for gauging the model's effectiveness in detecting positives. It is computed us-

ing the formula 2.12. Conversely, Specificity focuses on accurately recognizing negative instances, and it is calculated as per formula 2.13 [47].

Precision, another crucial metric, signifies the proportion of true positives among all positive predictions, aiding in assessing the accuracy of positive predictions made by the ML models 2.14. Lastly, Accuracy, computed according to formula 2.15, provides an overall assessment of the model's performance by measuring the proportion of total correct predictions, both positive and negative.

This comprehensive approach ensures a thorough and well-balanced evaluation of ML models, adhering to established standards in the field as indicated by prior research [47, 103].

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{2.12}$$

$$\text{Specificity} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}} \tag{2.13}$$

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{2.14}$$

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}} \tag{2.15}$$

The F1-Score is a key metric in machine learning, particularly important for blending Precision and Recall into a single, thorough measure, as cited in reference [21]. This combination is essential because using either Accuracy or Recall alone does not fully capture an algorithm's effectiveness. For example, a model might demonstrate high Precision but low Recall, signifying precise yet incomplete positive predictions. On the other hand, a model with high Recall but low Precision accurately identifies most positive cases and includes many false positives.

The F1-Score resolves this disparity by merging these two metrics into a cohesive score, as mentioned in reference [52]. It provides a more nuanced view of a model's performance, which is especially important in situations where bal-

ancing Precision and Recall is critical. The calculation of the F1-Score, detailed in formula 2.16, is especially useful in binary or multiclass classification tasks. It executes a calculation that considers both Precision and Recall, yielding a singular, comprehensive evaluation of an algorithm's effectiveness.

By amalgamating Precision and Recall, the F1-Score effectively acts as a dependable measure of a model's overall precision and thoroughness in predictions, proving to be a vital instrument for assessing machine learning algorithms.

$$\text{F1-Score} = {}^{2}\times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.16}$$

In multiclass classification issues, it's necessary to compute metrics like Accuracy, Recall, Precision, and the F1-Score for each class separately and then calculate their average. This method guarantees a complete evaluation of the model's effectiveness across all categories, adapting these metrics from their initial binary classification framework to fit the intricacies of multiclass situations.

# 3

# Methodology

This chapter provides a detailed exposition of the dataset curated for this research, encapsulating the collection, preparation, and structuration processes undertaken before deploying the data for the Integrative Deep Learning Framework aimed at cognitive load assessment (figure 3.1). This foundational stage is imperative as it ensures the data's cleanliness and compatibility, prerequisites for effectively evaluating our deep learning model. This methodology establishes the basis for precise and robust cognitive load measurement using ECG and pupil dilation data within behavioral studies.

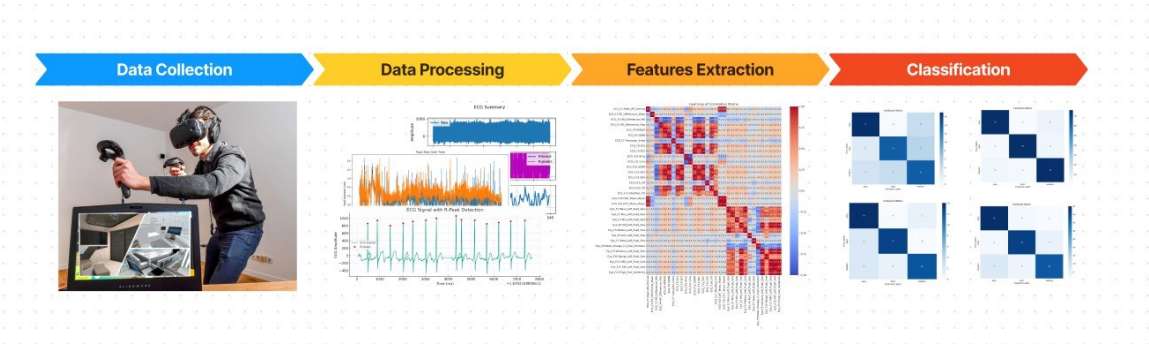


Figure 3.1: Stages of ECG and Eye Tracker Data Utilization for Cognitive Load Assessment

## 3.1 PARTICIPANTS

This research incorporated 21 individuals into a virtual reality (VR), carefully selected to span a broad demographic spectrum to ensure a detailed understanding of the VR experience. The average age of the participants was approximately 25.16 years, aligning with a young adult demographic that typically demonstrates adaptability and familiarity with innovative technologies such as VR.

The gender distribution of the participants was closely balanced, with 52.63% identifying as male and 42.11% as female. This distribution allowed the exploration of potential gender-specific dynamics within the VR environment, contributing valuable insights into the inclusive design of VR experiences.

Participants were selected based on criteria emphasizing diversity rather than specific skill sets or prior experience with VR. This inclusive approach ensured that the study's findings would apply to a general population relevant to various practical applications of VR, including training, education, and entertainment. The selection criteria were designed to eliminate biases that could influence the assessment of cognitive load and user experience in VR, thereby enhancing the validity of the research outcomes.

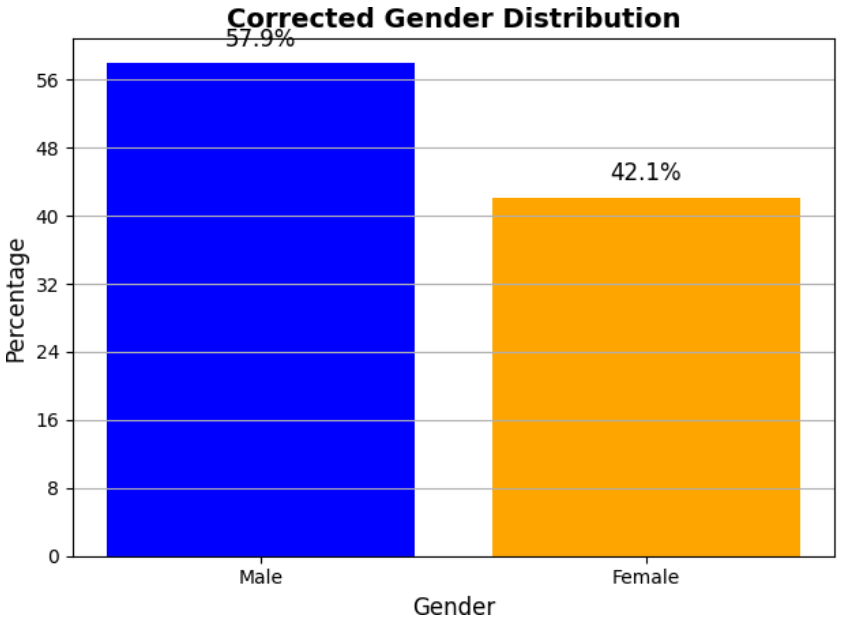


Figure 3.2: the corrected gender distribution of study participants

## 3.2 TASK DESCRIPTION

The virtual reality (VR) experiment was carefully crafted to assess cognitive load through participant engagement in dynamic, interactive tasks. Central to this evaluative process were memory-intensive tasks that required participants to classify colored packages on a moving conveyor belt. The experiment introduced a graduated difficulty structure, categorizing tasks into three ascending levels: 1-back, 2-back, and 3-back. This stratification necessitated the participants' recall and recognition of the colors of one, two, or three previously encountered packages. The experimental design manipulated two principal variables: the hue of the packages and the sequential memory burden, as indicated by the n-back level, to meticulously examine the effects of varying cognitive demands on participant performance.

## 3.3 EXPERIMENTAL PROCEDURE

The procedure of the experiment was delineated through a sequential process to ensure a structured and comprehensive assessment:

1. **Orientation:** Participants were initially acclimated to the VR environment in an orientation room. This space showcased calming visuals and collected baseline physiological measurements, preparing them for the tasks ahead.
2. **Introduction to Task Environment:** participants were transported to a virtual factory setting. In this environment, they were first introduced to the core task of sorting colored packages. The main task involved the sorting of colored packages.
3. **Practice Session:** Before the commencement of the main experiment, a 2-minute practice session was conducted. This session aimed to familiarize participants with the mechanics of the task, ensuring comfort and understanding.
4. **Main Experiment:** The participants then embarked on the main experiment, segmented into three consecutive task blocks—1-back, 2-back, and 3-back. Each block consisted of 30 stimuli, requiring sorting based on the assigned memory load.
5. **Feedback Mechanisms:** Participants received visual and auditory feedback throughout the experiment. This feedback was crucial in indicating the correctness of responses, with the factory environment's color and the audio clip's pitch altering to mirror performance levels.

6. **Cognitive Load Assessment:** Participants were asked to complete a questionnaire following each task block. This tool was designed to gauge the perceived cognitive load, offering insights into the mental effort expended during the tasks.
7. **Debrief and Data Collection:** The session culminated in a debriefing, where participants were informed about the experiment's aims and procedures. Additionally, psychophysiological data were collected for further analysis.

The procedure was carefully structured to ensure that participants could fully engage with the tasks without prior VR experience, allowing for an assessment of cognitive load indicative of a general population's interaction with VR technology.

## 3.4 DATA COLLECTION

The Unity environment, utilizing Editor version 2019.4.29, comprises two meticulously crafted scenes as depicted in Figure 3.3. The initial scene introduces users to a lobby room with a serene garden view alongside wall-mounted screens displaying calming imagery and relevant information as seen in a specific video segment[1] (0:00 min-0:17min). This introductory space is pivotal in orienting users to the Virtual Environment (VE) and establishing a baseline for physiological measurements. The importance of this baseline setting lies in its contribution to enhancing the accuracy of physiological data collection, particularly in contexts such as operator training and performance evaluation [45].

After the orientation phase, users are seamlessly transitioned to a virtual factory setting, which houses two critical apparatuses: a 3D printer and a conveyor belt [145]. The primary focus of this environment is the conveyor belt task, engineered to generate varying levels of cognitive load (CL). This task introduces participants to a series of colored packages (red, blue, purple, green, or yellow) [129], which advance towards the user on a conveyor belt. A sorting box is positioned within the VE to the user's right, and to their left is an additional conveyor belt. Users are tasked with transferring each package to the left conveyor belt if its color aligns with the package shown in the preceding n trials or to the sorting box. This task component is illustrated in a video segment[2] (1:41min-2:30min).

[1]https://www.youtube.com/watch?v=1vYoqcHMadU
[2]https://www.youtube.com/watch?v=1vYoqcHMadU

After a preliminary 2-minute practice session, users embark on three blocks of tasks - 1-back, 2-back, and 3-back, each comprising 30 stimuli. The assignment of package colors is randomized to maintain task variability. The duration of this task extends to 10 minutes [59].

The study utilizes a bifurcated feedback approach to enhance participant engagement and task performance, encompassing visual and auditory mechanisms. Visual feedback is manifested through the dynamic alteration of the factory floor's columns and ceiling colors, transitioning between green and red to reflect the user's task accuracy. Concurrently, auditory feedback is employed, featuring a pitch shift in a consistent audio clip to signify correct or incorrect responses, with the base pitch ranging from 0 to 1 and an offset increment of 0.1.

This comprehensive approach to data collection within the Unity environment facilitates cognitive load assessment and enriches the user's immersive experience through interactive tasks and multisensory feedback. The methodology outlined ensures a robust framework for examining the interplay between cognitive load and user interaction within VR settings.

Figure 3.3: The virtual environment.

To guarantee a smooth and integrated evaluation of cognitive load alongside participant feedback in the virtual environment (VE), the research protocol requires that after each task block, participants fill out a specifically designed six-question survey [54]. This instrument plays a crucial role in evaluating the perceived cognitive load, offering insights into the user's experience and the task's impact on their cognitive state. In parallel, psychophysiological data is meticulously acquired using advanced tools: the Polar H10 for electrocardio-

gram (ECG) measurements [16] and a Tobii eye tracker [17], which is seamlessly integrated into the head-mounted display (HMD) to monitor eye movements. Given the inherent differences in sampling rates among these varied data types, each source is meticulously recorded in its .csv file, dedicated to each specific task[3]. A bespoke Python class has been developed to streamline the processing and analysis of the gathered data. This class introduces a standardized processing pipeline designed to efficiently calculate features from the collected data, thereby producing separate data frames for each task. Accessibility and ease of use are enhanced through a user-friendly graphical user interface (GUI), which simplifies the process of data processing and facilitates the generation of output data systematically stored in a pre-specified folder. The culmination of this process results in the derivation of processed data that encompasses a comprehensive array of metrics. These metrics include performance measures, average scores from the NASA-TLX assessment of cognitive load, heart-related features, and analyzing the frequency of gaze switches [145]. This integrated approach ensures a cohesive and coherent framework for evaluating cognitive load and physiological responses, enhancing the study's overall efficacy and insight into the interactive dynamics within the VE.

## 3.5 DATASET PREPROCESSING

Our study executes the dataset preparation and preprocessing through a series of defined steps to ensure the data is ready for neural network model training. Initially, the dataset is loaded into a pandas DataFrame, a critical step for structuring the data for accessibility and manipulation. Subsequently, we filter out specific participants to align the dataset with our research parameters, maintaining the integrity and relevance of our study. Following this, feature scaling is applied using Scikit-learn's StandardScaler, a crucial process that normalizes the features' range across the dataset to aid the neural network's learning efficiency. Moreover, categorical labels indicating cognitive load levels—easy, medium, and hard—are transformed into numerical values via Scikit-learn's LabelEncoder. This transformation is vital for the neural network's algorithmic processing of these labels. To facilitate efficient batch processing, a cus-

[3]https://osf.io/754tu/?view$_o nly$ = $7f7d1b30affee4f99b3cd45bb631fcf91$

tom PyTorch CustomDataset class is developed to handle the preprocessed data, leveraging PyTorch's DataLoader for optimized computational efficiency. These steps ensure that the dataset is formatted correctly, normalized, and primed for practical neural network training.

### 3.5.1 PREPOSSESSING OF PHYSIOLOGICAL DATA

In this study, we employed a variety of Python libraries specialized in signal processing and analysis to preprocess and extract features from electrocardiogram (ECG) data. This included BioPSy for initial ECG signal processing, like filtering and R-peak detection, HRVAnalysis for removing ectopic beats from RR intervals, and PyWavelets for advanced wavelet analysis. Additionally, SciPy was integral for various signal-processing tasks and statistical calculations, including using its Short-Time Fourier Transform (STFT) and Welch frequency analysis method and its statistical functions for deriving measures like kurtosis and skew.

## PREPROCESSING AND FREQUENCY DOMAIN ANALYSIS OF ECG DATA

The preprocessing of ECG data begins with extracting and processing the signal at a sampling rate of 130 Hz. To isolate R-peaks, a bandpass filter within the range of 0.5 to 50 Hz is employed to eliminate extraneous noise, succeeded by applying the EngZee segmenter algorithm [90]. Subsequently, RR intervals are determined from these detected peaks, with ectopic beats being excised to ensure a precise analysis of heart rhythm.

The subsequent phase involves the frequency domain function analyzing the heart rate variability (HRV) within the frequency domain based on these RR intervals. This analysis initiates with the interpolation of RR intervals to achieve uniform spacing, setting the stage for the accurate execution of the Welch method. This method is tasked with determining the power spectral density of the intervals, dividing the data into segments with a maximum length of 256 points. It then delineates the low-frequency (0.04 to 0.15 Hz) and high-frequency (0.15 to 0.4 Hz) bands, essential for deducing HRV metrics. The power within these frequency bands is quantified using the trapezoidal rule, and the LF/HF ratio is calculated, serving as an essential marker for assessing the autonomic

nervous system's equilibrium. This comprehensive process ultimately yields the LF and HF power values and their ratio, providing an in-depth analysis of cardiac rhythm and autonomic nervous system interactions based on the ECG data.

Table 3.1: Summary of ECG Data Preprocessing and HRV Analysis Steps

| **Process Step** | **Details** |
|---|---|
| ECG Sampling Frequency | 130 Hz |
| Bandpass Filter | 0.5 to 50 Hz (to remove noise) |
| R-peak Detection | EngZee segmenter algorithm |
| RR Interval Calculation | From R-peaks |
| Ectopic Beat Removal | For accurate heart rhythm analysis |
| HRV Frequency Domain Analysis | |
| - Interpolation | Uniform spacing of RR intervals |
| - Welch Method | Power spectral density computation |
| - Segment Length | Up to 256 points |
| - Frequency Bands | Low-frequency (0.04 to 0.15 Hz)<br>High-frequency (0.15 to 0.4 Hz) |
| - Power Calculation | Trapezoidal method |
| - LF/HF Ratio | Autonomic nervous system balance indicator |
| Output | LF power, HF power, LF/HF ratio |

### 3.5.2 LABELING OF PHYSIOLOGICAL DATA

In experimental research, precisely labeling physiological data is crucial for mapping participant responses to specific experimental conditions or occurrences. This segment of our Python script demonstrates a methodical approach to annotating ECG (electrocardiogram) and eye-tracking data, ensuring alignment with various study phases. The initial step in this labeling process involves identifying critical events within the study. These events, which could range from 'baseline' periods to 'practice' sessions or different levels of task difficulty, are meticulously recorded with accurate timestamps in an 'events.csv' file. This file's timestamps are vital for accurately labeling each participant's dataset.

To maintain uniformity across the datasets, the script utilizes a function, clean_col_name, to standardize the column names in the ECG and eye-tracking data files. This function converts all text to lowercase, strips unnecessary spaces or special characters, and ensures that column names are consistent, clear, and error-free. Furthermore, when an event recurs (for example, multiple 'practice'

sessions), the script employs an intelligent mechanism to distinguish each occurrence by appending an index (such as 'practice_0', 'practice_1'). This nuanced approach to labeling is indispensable in experimental designs where participants are subjected to repeated or similar conditions, providing a structured and error-free dataset for subsequent analysis.

**Algorithm 1** EngZee Segmenter for R-peak Detection

**Require:** ECG signal $S$, Sampling rate $f_s$
**Ensure:** R-peak indices

**1. Preprocessing:**
2. Apply a bandpass filter to $S$ with cutoff frequencies $f_{low} = 8\,Hz$ and $f_{high} = 20\,Hz$.
3. Derive the signal to emphasize the QRS complex: $S' = \frac{dS}{dt}$.
**4. Squaring and Integration:**
5. Square the derivative: $S'' = (S')^2$.
6. Apply a moving window integration to $S''$ with a window width $W$, typically related to the expected width of the QRS complex.
**7. Thresholding and Peak Detection:**
8. Set a threshold, $T$, based on the median or mean of $S''$.
9. Identify segments where $S''$ exceeds $T$.
10. Within each segment, locate the local maximum, which is considered an R-peak candidate.
**11. Post-Processing:**
12. Apply adaptive thresholding to distinguish true R-peaks from noise.
13. Use a search-back algorithm to correct for missed R-peaks, adjusting the threshold as needed. RETURN Indices of detected R-peaks.

## SEGMENTING DATA BASED ON EVENTS

The core of the labeling process is segmenting the physiological data according to these events. The script uses the pandas.cut() function segments the ECG and eye-tracking data based on the event timestamps, assigning each data point a label corresponding to the ongoing event. This segmentation is integral for analyzing physiological responses in specific experimental conditions. Finally, the script saves each participant's labeled ECG and eye-tracking data. This data, now segmented and labeled in line with the study's events, is ready for detailed analysis, offering researchers insights into how different conditions affect physiological responses. The automated labeling of physiological data in this script

not only streamlines the data processing workflow but also enhances the accuracy and reliability of the analysis. By aligning physiological responses with specific events in an experiment, researchers can draw more precise and meaningful conclusions, crucial in psychology, neuroscience, and human-computer interaction.

## 3.6 FEATURE CALCULATION

Feature extraction is increasingly critical in analyzing biomedical signals, mainly as datasets grow to include thousands of features. This surge in data availability results from the extended periods over which biomedical signals can now be collected, introducing unique challenges to the field [68, 96]. Understanding physiological signal properties is foundational for effective analysis. These signals are characterized by several distinct features [68]:

A. **Non-stationary:** Their properties change over time.

B. **Non-linear:** Their behavior is not captured by linear models.

C. **Non-Gaussian:** They do not follow a normal distribution.

D. **Non-short form:** They extend over lengthy periods, complicating analysis.

These characteristics add complexity to feature extraction and signal analysis, emphasizing the need for extracted features to capture relevant signal patterns and behaviors precisely [68, 96]. Before feature extraction, it is crucial to convert continuous analog signals to discrete digital signals using an analog-to-digital converter (ADC). This step is essential for pattern recognition across discrete intervals [68]. Post-extraction, feature selection plays a key role. Selecting the right features for machine learning model training can significantly affect the model's performance, positively or negatively.

### 3.6.1 ECG FEATURES

The electrocardiogram (ECG) signal is a rich source of information regarding cardiac function, offering critical insights that can be unlocked through feature extraction processes [64]. ECG features include frequency, time, morphology, energy, and the RR interval, each providing a different perspective on the heart's electrical activity [83]. The extraction of these features involves various computational techniques meticulously designed to quantify distinct elements of the

ECG signal [109]. For practical utility, it is advisable to tailor feature selection to the specific requirements of the application at hand. This ensures the chosen features are directly relevant to the patterns and behaviors of interest, facilitating a more focused and effective analysis [96, 68].

## THE KEY STATISTICAL OF TIME-DOMAIN

A key aspect of feature extraction within ECG data analysis focuses on time-domain measurements, which are pivotal for evaluating heart rate variability (HRV). These time-domain features encompass statistical measures such as the mean, median, standard deviation, skewness, and kurtosis of the RR intervals, offering a comprehensive view of the heart's rhythmic patterns [56, 64]. Further enriching this analysis, including the RMSSD (Root et al. of Successive Differences) metric provides a nuanced measure of HRV, capturing the variability in time intervals between successive heartbeats [12].

**The formula for RMSSD is:**

$$RMSSD = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N-1}(RR_{i+1} - RR_i)^2} \tag{3.1}$$

In this formula:

- $RR_i$ and $RR_{i+1}$ represent successive RR intervals.
- $N$ is the total number of RR intervals.
- The term $(RR_{i+1} - RR_i)^2$ calculates the square of the difference between successive RR intervals.
- The sum of these squared differences is then divided by $N - 1$, which is the total number of intervals minus one.
- Finally, the square root of this value gives the RMSSD.

Features like the mean and median are used to assess the central tendency of the ECG signal, while the standard deviation, range, and interquartile range capture its statistical dispersion. Kurtosis and skewness are employed to evaluate the asymmetry and peak sharpness of the ECG signal distribution [6].

Table 3.2: Time Domain Features

| Parameter | Units | Description |
|---|---|---|
| SDRR | ms | Standard Deviation of RR intervals. |
| NNx | Count | Number of pairs of successive RR intervals. |
| pNNx | % | Proportion of NNx intervals. |
| SD1 | ms | Standard deviation, measuring short-term HRV. |
| SD2 | ms | Standard deviation, measuring long-term HRV. |
| CSI | Unitless | Cardiac Sympathetic Index. |
| CVI | Unitless | Cardiac Vagal Index, a logarithmic transformation. |

# THE KEY STATISTICAL OF FREQUENCY-DOMAIN

In the realm of ECG signal analysis, frequency-domain features are indispensable for a detailed examination of heart rate variability (HRV) and for elucidating the interplay between sympathetic and parasympathetic nervous systems in modulating heart rate dynamics[84]. The Power Spectral Density (PSD) analysis is a fundamental tool in this context, dissecting the ECG signal to isolate and examine its frequency constituents. This analysis illuminates the critical segments of HRV across the very-low-frequency (VLF), low-frequency (LF), and high-frequency (HF) bands. These frequency bands play a pivotal role in decoding the nuances of autonomic nervous system regulation and comprehensively understanding the breadth of HRV[84].

**Power Spectral Density (PSD) formula:**

$$PSD(f) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{MW} |X_i(f)|^2 \tag{3.2}$$

where:

- $PSD(f)$ is the Power Spectral Density at frequency $f$.
- $N$ is the number of segments the signal is divided into.
- $M$ is the length of each segment.
- $W$ is the normalization factor, calculated as $W = \sum w(t)^2$, the sum of the squared window function.
- $X_i(f)$ is the Fast Fourier Transform (FFT) of the $i$-th windowed segment of the signal.

Table 3.3: Frequency Domain Features

| Parameter | Units | Description |
|---|---|---|
| VLF Power | $ms^2$ | Power in the Very Low-Frequency range (0.003-0.04 Hz). |
| LF Power | $ms^2$ | Power in the Low-Frequency range (0.04-0.15 Hz). |
| HF Power | $ms^2$ | Power in the High-Frequency range (0.15-0.4 Hz). |
| Total Power | $ms^2$ | Total spectral power across all frequency bands. |
| LF/HF Ratio | Unitless | Ratio of LF to HF power, a measure of autonomic balance. |
| Spectral Entropy | Unitless | Entropy of the power spectral density. |
| Normalized LF | % | LF power normalized by total power minus VLF power. |
| Normalized HF | % | HF power normalized by total power minus VLF power. |

### 3.6.2 PUPIL DILATION FEATURES

In the virtual reality (VR) technology[19], integrating eye-tracking sensors is a critical advancement. These sensors are adept at monitoring changes in pupil dilation within VR environments, thus providing a pathway to unearth valuable insights regarding users' emotional responses to VR content. A notable example is the Pupil Labs HTC eye-tracker, distinguished by its eye gaze tracking capability of 120 Hz, a high accuracy of 0.6 degrees, and the precision to measure pupil diameter to within 1 mm [23]. Similarly, Varjo's eye-tracking technology proves instrumental in gathering data on user fixations, saccades blink, and pupil dilations [40]. Varjo extends this functionality by incorporating pupil dilation measurements, enriching its application in research and social VR settings [42]. Research findings have illustrated that eye fixation, utilized as a metric for learning, enhances the accuracy of emotion classification within VR environments, offering superior performance over pupil diameter measurements in this regard [146]. In this context, our study leverages computational methods to dissect eye-tracking data, mainly focusing on pupil dilation metrics, focusing predominantly on the left pupil diameter. Within the statistical summary in Table**??**, a comparative analysis reveals the descriptive statistics about the pupil diameters of both the left and right eyes. This analysis highlights a notable consistency across various statistical metrics, affirming the symmetrical nature of pupil responses to stimuli. Nonetheless, the analysis also identifies minor discrepancies, which warrant further exploration.

In our research, we delve into various statistical metrics to understand the dynamics of pupil dilation. These metrics indicate the eye's response over time and under different conditions.

Table 3.4: Transposed Descriptive Statistics

| Statistic | Right.PupilDiameter | Left.PupilDiameter |
|---|---|---|
| Valid | 65437 | 65437 |
| Missing | 0 | 0 |
| Mean | 3.344 | 3.067 |
| Std. Deviation | 1.147 | 1.142 |
| Coefficient of Variation | 0.343 | 0.372 |
| Variance | 1.317 | 1.305 |
| Range | 6.669 | 6.669 |
| Minimum | -1.000 | -1.000 |
| Maximum | 5.669 | 5.669 |

# STATISTICAL ANALYSIS OF PUPIL SIZE

**Kurtosis:** These statistical tests shed light on the shape and asymmetry of the pupil size distribution. Kurtosis is an indicator of the extent to which a distribution's peak deviates from that of a normal distribution. A distribution with a high kurtosis (a value exceeding +2) is more sharply peaked, suggesting a concentration of responses in the middle. Conversely, a kurtosis value lower than -2 signals an overly flat distribution or spread out [74, 106].

**Formula**

$$Kurtosis = \frac{n(n+1)}{(n-1)(n-2)(n-3)} \sum_{i=1}^{n} \left( \frac{x_i - \text{Mean}}{\text{Standard Deviation}} \right)^4 - \frac{3(n-1)^2}{(n-2)(n-3)} \tag{3.3}$$

Where:
$n$ = number of observations
$x_i$ = each individual data point
$\bar{x}$ = mean of the data
$s$ = standard deviation of the data

**Interquartile Range (IQR):** The IQR focuses on the middle 50% of pupil sizes, offering insights into the central tendency of pupil dilation while eliminating the influence of outliers.

The interquartile range (IQR) is calculated as:

$$IQR = Q_3 - Q_1 \tag{3.4}$$

**Area Under the Curve (AUC):** The AUC provides a singular value encapsulating the overall pupil dilation experience. It combines the size and duration of dilation into one metric, offering a unique and comprehensive perspective on the eye's response over time.

The area under the curve (AUC), assuming equally spaced measurements, is:

$$AUC = \sum_{i=1}^{n-1} \frac{(x_i + x_{i+1})}{2} \times \Delta t \tag{3.5}$$

These diverse metrics offer a detailed and nuanced understanding of pupil dilation, which is vital for interpreting physiological responses in our study. They help correlate pupil behaviour with various stimuli or psychological states, enhancing the depth and breadth of our research findings.

Table 3.5: Analysis of Pupil Size Features

| Feature | Units | Description |
|---|---|---|
| Mean Pupil Diameter | mm | The average pupil diameter. |
| Standard Deviation | mm | Variation in pupil diameter size. |
| Max Pupil Diameter | mm | The largest recorded. |
| Min Pupil Diameter | mm | The smallest recorded. |
| Amplitude of Pupil Oscillations | mm | The difference between max and min pupil diameters |
| Kurtosis | Unitless | Sharpness of the distribution. |
| Skewness | Unitless | Asymmetry of the distribution. |

## 3.7 FEATURE SELECTION USING FISHER SCORES

The Fisher score is employed as a strategy for feature selection by ranking features based on their ability to differentiate between distinct classes or groups. This method, rooted in a filter-based, supervised feature selection framework, allocates weights to features, with a higher Fisher score denoting a feature's enhanced discriminative capability and its value in classification endeavours [3, 116].

### THE FISHER SCORE MODEL

The fundamental concept of the Fisher Score involves selecting a group of features [119] that effectively maximizes the distinction between data points be-

longing to different classes and, at the same time, minimizes the distinction within the same class. This process involves calculating a score for every feature to determine its ability to discriminate between classes. In particular, given a training dataset $X \in R^{m \times n}$ with respect to $c$ different classes, the Fisher score of the $i^{th}$ feature is computed [139, 117] by

$$FS(f_i) = \frac{S_B(f_i)}{\sum_{k=1}^{c} S_{W_k}(f_i)}, \tag{3.6}$$

where $S_B(f_i) = \sum_{k=1}^{c} n_k(\mu_i^{(k)} - \mu_i)^2$ is the between-class scatter of the $i^{th}$ feature, $n_k$ is the number of samples in the $k^{th}$ class, $\mu_i^{(k)}$ is the mean of the $i^{th}$ feature in the $k^{th}$ class, $\mu_i$ is the mean of the $i^{th}$ feature in X, $S_W(f_i) = \sum_{j=1}^{n_k}(x_{ij}^{(k)} - \mu_i^{(k)})^2$ is the within-class scatter matrix of the $i^{th}$ feature with respect to the $k^{th}$ class, and $x_{ij}^{(k)}$ denotes the value of the $i^{th}$ feature for the $j^{th}$ sample in the $k^{th}$ class.

## THE FISHER SCORE IN CLASSIFYING COGNITIVE LOAD STATES

The Fisher score significantly contributes to the feature selection process in developing advanced deep learning models for classifying cognitive load states [111]. Its adoption within our research framework serves multiple crucial purposes: it facilitates the practical distinction between classes, reduces the complexity of the dataset, improves model performance, illuminates principal physiological indicators of cognitive load, and ensures methodological rigor [111]. This score is pivotal in evaluating the discriminative capacity of individual features across varying cognitive load states. By assigning priority to features according to their Fisher scores, our approach refines the dataset by eliminating unnecessary noise and less pertinent features. Concentrating on the most discriminative features, as highlighted by their elevated Fisher scores, generally yields models that are not only more precise but also exhibit superior generalization capabilities, thereby achieving a harmonious balance between complexity and practicality [117, 111]. Moreover, the feature selection process via the Fisher score reveals critical physiological markers of cognitive load, deepening our comprehension of cognitive load evaluation [102, 118]. Utilizing the

Fisher score in our research lays a quantifiable, statistically sound basis for feature selection, guaranteeing that model development is empirically solid and data-driven [118].

```
def calculate_fisher_scores(X, y):
    selector = SelectKBest(score_func=f_classif, k='all')
    selector.fit(X, y)
    scores = selector.scores_
    return scores

# Calculate Fisher scores for features
fisher_scores = calculate_fisher_scores(df[feature_columns], df['
    label_encoded'])

# Copy the original feature columns
original_feature_columns = feature_columns.copy()

# Selecting the top features based on Fisher scores
num_top_features = 15  # Select the top 15 features
top_features_indices = np.argsort(fisher_scores)[-num_top_features:]
top_features = [feature_columns[i] for i in top_features_indices]

# Update the feature columns to include only the top selected
    features
feature_columns = top_features
```

Code 3.1: Feature Selection using Fisher Scores

The accompanying figure 3.4 illustrates the top 15 features selected based on their Fisher Scores from the ECG and pupil dilation datasets. It is evident from the chart that 'Pupil Size' has the highest Fisher Score, signifying its discriminative solid ability in the context of cognitive load classification. The chart provides a clear visualization of the feature selection process, showcasing the scores from highest to lowest and thus highlighting the most informative features for our deep learning models. This prioritization of features is instrumental in enhancing the accuracy and efficiency of our classification tasks.

Table 3.6: Top 15 Features based on Fisher Scores

| Rank | Feature | Fisher Score (Approx.) |
|---|---|---|
| 1 | Pupil Size | 0.1 |
| 2 | Pupil Size Std | 0.15 |
| 3 | Band Power | 0.2 |
| 4 | LF/HF Ratio | 0.25 |
| 5 | Modified CVI | 0.3 |
| 6 | LF Norm | 0.35 |
| 7 | HF Norm | 0.4 |
| 8 | ECG F5 SDNN | 0.45 |
| 9 | ECG F2 Differences Mean | 0.5 |
| 10 | ECG F3 Differences Max | 0.55 |
| 11 | ECG F4 SD1 | 0.6 |
| 12 | ECG F9 SD2 | 0.65 |
| 13 | ECG F11 LFnu | 0.7 |
| 14 | ECG F10 HFnu | 0.75 |
| 15 | ECG F19 Total Power | 2.5 |

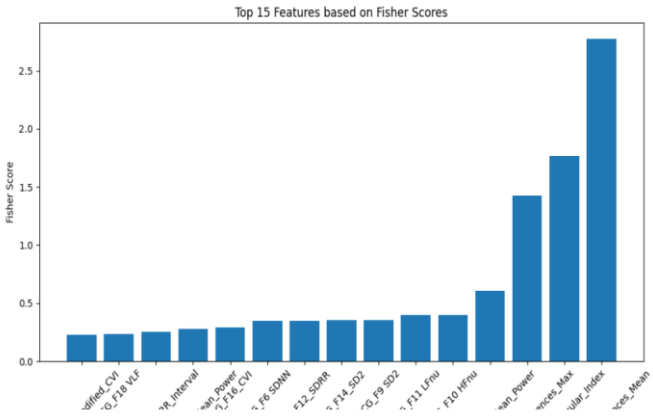


Figure 3.4: Feature selection with Fisher score

## 3.8 ARTIFICIAL NEURAL NETWORKS

This research investigated cognitive load classification by deploying two neural network architectures: the Multilayer Perceptron (MLP) and the Gated Recurrent Unit (GRU). Each model was chosen for its unique strengths in handling different data types and contributing to our understanding of cognitive load dynamics.

### 3.8.1 MLP

In our pioneering journey to understand human behaviour through the lens of cognitive load states, we employed cutting-edge Deep Learning models. Specifically, we utilized the Multi-Layer Perceptron (MLP) to classify cognitive load states by integrating ECG and pupil dilation data. This approach allowed us to delve deep into the nuances of how cognitive load impacts physiological and psychological responses. The MLP is a cornerstone in our methodology thanks to its robust framework for managing complex, multi-dimensional data. It is a configuration that excels in pattern recognition and is ideally suited for our purpose. The MLP architecture we designed includes several fully connected layers, or linear layers, which are the bedrock of the model. These layers perform essential linear transformations on the input data, mapping it efficiently to an output that correlates with different cognitive load states. We enriched the MLP with batch normalization layers to ensure consistent data distribution throughout the network. This step is crucial for maintaining the speed and stability of the training process. It addresses potential issues of internal covariate shift, paving the way for faster model convergence. Non-linearity is injected into the MLP through ReLU (Rectified Linear Unit) activation functions. This choice allows our model to capture and represent complex patterns in the data, a capability beyond the reach of linear models. ReLU is particularly valued for its effectiveness and simplicity in introducing non-linearity into our model. Dropout layers were strategically included to enhance the model's robustness and prevent overfitting. These layers randomly deactivate specific neurons during training, forcing the network to learn more generalized features that are not dependent on minor data points. This technique ensures that our model is accurate and reliable when applied to new, unseen data.

```
1
2  class MLP(nn.Module):
3      def __init__(self, input_size, output_size, dropout_rate=[0.5,
       0.6], activation='relu'):
4          super(MLP, self).__init__()
5          self.fc1 = nn.Linear(input_size, 64)
6          self.bn1 = nn.BatchNorm1d(64)
7          self.relu1 = nn.ReLU()
8          self.dropout1 = nn.Dropout(dropout_rate[0])
9          self.fc2 = nn.Linear(64, 16)
10         self.bn2 = nn.BatchNorm1d(16)
```

```
        self.relu2 = nn.ReLU()
        self.dropout2 = nn.Dropout(dropout_rate[1])
        self.fc3 = nn.Linear(16, output_size)
```

Code 3.2: MLP Model

### 3.8.2 GRU

Diving into the intricacies of human behavior and cognitive load through ECG and pupil dilation data, we have also embraced the power of the Gated Recurrent Unit (GRU) model. Our GRU model is designed with precision to harness time-series data effectively, a perfect match for the temporal nature of our study. We configured the GRU with a focus on simplicity and efficacy. It features a singular GRU layer chosen to streamline the model's architecture without compromising its capability to process complex sequences. This decision reduces the model's complexity, making it more manageable and less prone to overfitting. Acknowledging the potential risk of overfitting, we have implemented an increased dropout rate. This strategy introduces randomness into the model training process, effectively helping the network generalize better by preventing it from relying too heavily on any specific data part. Our GRU model operates bidirectionally to efficiently capture the nuances of time-series data. This means it processes data forwards and backward, providing a more comprehensive understanding of the dataset. The output from this bidirectional GRU feeds into a simplified, fully connected layer designed to output the classification results based on the processed ECG and pupil dilation data. By choosing a bidirectional approach, we ensure our model captures all temporal dependencies, offering a richer, more nuanced analysis of cognitive load states. This GRU model is a testament to our commitment to leveraging deep learning in innovative ways to illuminate the subtle dynamics of human behavior.

```
class GRUModel(nn.Module):
    def __init__(self, input_size, hidden_size, output_size,
    num_layers=1, dropout_rate=0.5):
        super(GRUModel, self).__init__()

        # Using a single GRU layer instead of multiple to reduce
    complexity
        self.gru = nn.GRU(input_size, hidden_size, num_layers=
    num_layers,
```

```
                batch_first=True, bidirectional=True)

        # Increasing dropout to prevent overfitting
        self.dropout = nn.Dropout(dropout_rate)

        # Simplifying the fully connected layer
        self.fc = nn.Linear(hidden_size * 2, output_size) # *2 for
    bidirectional

    def forward(self, x):
        x, _ = self.gru(x)
        x = self.dropout(x)
        x = x[:, -1, :]    # Taking the output of the last time step
        x = self.fc(x)
        return x
```

Code 3.3: GRU Model Class Definition

## 3.9 MACHINE LEARNING CLASSIFIERS IN STACKED ENSEMBLE MODEL

Our approach employs a stacked ensemble model, integrating several machine learning classifiers to capitalize on their individual and combined strengths for enhanced classification accuracy of cognitive loads. The ensemble model includes:

- **Random Forest (RF)**, an ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting the mode of the classes (classification) of the individual trees. RF's resilience against overfitting and its aptitude for handling high-dimensional data makes it a valuable component of our model.
- **Support Vector Classifier (SVC)**, known for its effectiveness in high-dimensional spaces, employs a hyperplane to distinguish between classes, offering robustness to the model.
- **Light Gradient Boosting Machine (LightGBM)** is a gradient boosting framework that utilizes tree-based learning algorithms. It is recognized for its efficiency and performance in processing large datasets and categorical features.

The essence of our stacked ensemble model lies in its hierarchical architecture. The primary tier consists of the base classifiers, functioning collectively yet independently, to provide preliminary predictions. The subsequent tier, or the

meta-classifier, synthesizes these initial predictions into a consolidated output. Through logistic regression, this meta-classifier is adept at weighing the probabilistic contributions of each base model, ensuring a comprehensive analysis that accounts for the nuanced variations across input features. The ensemble model chosen structure indicates our commitment to a methodology that not only captures the intricacies of the data but also amplifies the predictive accuracy through strategic classifier integration.

### ADDRESSING DATASET IMBALANCES

Acknowledging the prevalence of class imbalances in physiological datasets pertinent to cognitive loads, our model incorporates the Synthetic Minority Oversampling Technique (SMOTE). This intervention augments the representation of minority classes within the training dataset, thus rectifying imbalances and enhancing the model's ability to generalize across diverse data points.

## 3.10 TECHNIQUES TO MITIGATE OVERFITTING

In supervised machine learning, a prevalent issue is the challenge of overfitting, where models excellently adapt to training data but stumble when applied to new, unseen datasets [37]. This phenomenon, recognized for its detrimental impact on a model's ability to generalize, is marked by a model's flawless performance on its training set contrasted with its underperformance on the test set. Overfitting stems from a model's tendency to learn and memorize the training data, capturing its noise as patterns, which skews its predictive capabilities [142].

Our exploration adopts two acclaimed strategies to mitigate this challenge: Dropout and Batch Normalization. These methods are integral to improving model performance and reducing the risk of overfitting, thereby ensuring that our models learn effectively from training data and retain their accuracy on novel datasets [32].

Table 3.7: Overfitting Prevention Techniques Utilized in MLP and GRU Models

| **Model** | **Dropout** | **L-Regularization** | **BN** | **Optuna** | **Data Standardization** |
|---|---|---|---|---|---|
| MLP | ✓ | ✓ | ✓ | ✓ | ✓ |
| GRU | ✓ | ✓ | | ✓ | ✓ |

### 3.10.1 DROPOUT

Dropout, a cutting-edge regularization strategy, stands apart from traditional regularization methods like L1 and L2, which typically adjust the cost function. Instead, dropout transforms the architecture of the network itself. This strategy is particularly relevant in the context of deep neural networks. With their multiple non-linear hidden layers, these networks can learn complex functions. However, they are susceptible to overfitting, especially when training data is sparse. Dropout, as a regularization technique, is crafted to address this very challenge [33].

The core mechanism of dropout involves the random exclusion of neurons and their connections from the neural network during each training iteration. This ingenious process emulates the effect of training numerous distinct neural networks, echoing the principles behind ensemble methods. While each network may overfit in its unique manner, collectively, they work to counterbalance their respective overfitting tendencies. A notable feature of dropout is the shared weights across these various network configurations, leading to infrequent training for each specific configuration. However, this approach has demonstrated significant efficacy in meeting regularization objectives. [112]. The advent of dropout in 2014 marked a pivotal moment in regularization technology. Its implementation showcased an improvement in classification accuracy on the MNIST dataset by 0.4%, particularly when combined with L2 regularization. Since its introduction, dropout's ability to diminish overfitting has gained widespread acknowledgment in a plethora of applications. By compelling networks to not rely excessively on any single feature, dropout fosters the learning of more robust features, thus enhancing the model's generalizability and reliability across different settings. This narrative underscores the foundational contributions of Srivastava et al. (2014) in introducing dropout as a robust solution to overfitting, further solidified by the empirical evidence presented in their seminal work [112].

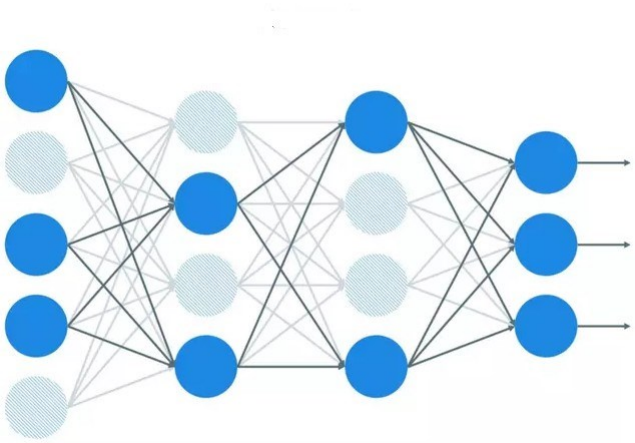

Figure 3.5: Dropout

### 3.10.2 BATCH NORMALIZATION

Batch normalization (BatchNorm), introduced by Ioffe and Szegedy in 2015, has revolutionized the architecture of deep learning models [60]. This technique is a cornerstone in supervised learning for standardizing the outputs across the layers within a neural network, essentially "resetting" the distribution of these outputs before they proceed to the next layer. This reset improves the subsequent layer's ability to interpret the data more efficiently, facilitating a smoother and more effective learning process [136].

The implementation of BatchNorm brings a multitude of benefits to the neural network training procedure. Primarily, it stabilizes the training by addressing the internal covariate shift, a common challenge during the training phase. This shift often complicates the model optimization, but BatchNorm simplifies this by ensuring a consistent distribution of inputs across the layers. Moreover, it significantly bolsters the model's generalization capabilities. By normalizing the activations in each layer, BatchNorm reduces the risk of overfitting, thereby improving the model's performance on unseen data. These advantages underscore the efficacy of BatchNorm in enhancing both the performance and efficiency of neural network training [136].

Fundamentally, BatchNorm stabilizes the input distribution for each network layer during training, especially over a minibatch. It achieves this by introducing additional layers that adjust the distribution's first two moments—mean and variance—to zero and one, respectively. Moreover, inputs undergoing batch

normalization may be scaled and shifted via trainable parameters, preserving the network's ability to express complex functions. Typically, this normalization precedes the prior layer's activation function, ensuring optimal data preparation for the following processing stage [105].

During training, the activations of a layer are normalized for each mini-batch of data using the following equations:

Mean Calculation:

$$\text{mean} = \frac{1}{m} \sum_{i=1}^{m} x_i$$

Variance Calculation:

$$\text{variance} = \frac{1}{m} \sum_{i=1}^{m} (x_i - \text{mean})^2$$

Normalization of Activations:

$$y_i = \frac{(x_i - \text{mean})}{\sqrt{\text{variance} + \varepsilon}}$$

Scaling and Shifting of Normalized Activations:

$$z_i = \gamma y_i + \beta$$

where $\gamma$ and $\beta$ are learned parameters.

During inference, the activations of a layer are normalized using the mean and variance of the activations calculated during training, rather than using the mean and variance of the mini-batch:

Normalized Activations:

$$y_i = \frac{(x_i - \text{mean})}{\sqrt{\text{variance} + \varepsilon}}$$

Scaled and Shifted Activations:

$$z_i = \gamma y_i + \beta$$

### 3.10.3 REGULARIZATION AND CLASS WEIGHT HANDLING

In deep learning, embracing regularization and adeptly managing class weights are crucial for augmenting model efficacy and minimizing the risk of overfitting [57, 78].

#### REGULARIZATION

Regularization enhances a model's generalizability and curtails overfitting. This is accomplished by introducing methods that balance the reduction of generalization error through a delicate trade-off—increasing bias to decrease variance. Among the spectrum of regularization techniques, L1 and L2 regularizations are noteworthy. Both methodologies augment the loss function with a penalty term designed to restrain the magnitude of the weights, thereby preventing the model from overfitting to the training data [63, 130, 100].

## L1 REGULARIZATION (LASSO)

LASSO regression, through the use of genetic algorithm descriptors, effectively simplifies model complexity by reducing coefficients, as first detailed by Tibshirani in 1996 [71]. It combines subset selection benefits with ridge regression's stability, aiming to minimize the Mean Squared Error (MSE) and the predictor count. The optimal coefficient set, $b_{\text{LASSO}}$, is determined at the point of MSE minimization, influenced by the tuning parameter $\lambda$, where a larger $\lambda$ induces more zero-value coefficients [85, 70, 130].

**L1 formula** where the objective is to minimize the residual sum of squares combined with the L1 norm of the coefficients, the equation can be denoted[30] as:

$$\hat{\beta}_{\text{LASSO}} = \arg\min_{\beta} \left( \frac{1}{2n} \sum_{i=1} (y_i - X_i\beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right). \tag{3.7}$$

## L2 REGULARIZATION( RIDGE REGULARIZATION)

The L2 regularization formula, commonly used to penalize the magnitude of coefficients in regression models, is given by [99, 85]:

$$\mathcal{L}_{L2} = \mathcal{L} + \lambda \sum_{i=1}^{n} w_i^2, \tag{3.8}$$

Where:

- $\mathcal{L}$ is the original loss function of the model.
- $\lambda$ is the regularization parameter.
- $w_i$ are the coefficients of the model.
- The sum is taken over all $n$ coefficients.

### 3.10.4 EARLY STOPPING TECHNIQUE

In our deep learning model, we embraced an essential regularization technique called EarlyStoppingto to ward off overfitting. During our GRU model's training phase, Early Stopping was pivotal, optimizing both performance and efficiency.

This regularization method is key to combating one of machine learning and deep learning's frequent hurdles: overfitting, which happens when a model overly familiarizes itself with the training data, learning its noise and fluctuations at the cost of its ability to perform well on new, unseen data. The EarlyStopping class offers a robust remedy for this issue, ensuring our model remains generalizable and effective [73, 17].

The EarlyStopping class is initialized with four key parameters:

1. **Patience (patience=7)**: The patience parameter in our EarlyStopping setup waits for seven epochs without improvement in validation loss before halting training. This ensures efficiency by stopping training early if progress stalls.
2. **Verbose (verbose=False)**:
3. **Minimum Change (delta=0)**: The delta parameter sets the minimum change in the monitored quantity to qualify as an improvement. This prevents stopping training for very minute improvements.
4. **Checkpoint Path (path='checkpoint.pt')**: This parameter specifies the path to save the model when a new minimum in the validation loss is observed. This ensures that the model is preserved at its best performance state.

The EarlyStopping method operates by continuously monitoring the validation loss after each epoch. Training is halted if the validation loss does not improve beyond the specified delta for several epochs equal to the patience value. This technique ensures that the model does not overfit and maintains a generalized performance.

Additionally, the EarlyStopping class includes functions to save the model checkpoint (save_checkpoint) when a new minimum validation loss is achieved and to load the best model (load_best_model) for future use or evaluation. This functionality is critical in our deep learning workflow, ensuring the retention and availability of the most effective version of the model.

```
class EarlyStopping:
    def __init__(self, patience=7, verbose=False, delta=0, path='checkpoint.pt'):
        self.patience = patience
        self.verbose = verbose
        self.counter = 0
        self.best_score = None
        self.early_stop = False
        self.val_loss_min = np.Inf
        self.delta = delta
        self.path = path

    def __call__(self, val_loss, model):
        score = -val_loss

        if self.best_score is None:
            self.best_score = score
            self.save_checkpoint(val_loss, model)
        elif score < self.best_score + self.delta:
            self.counter += 1
            if self.verbose:
                print(f'EarlyStopping counter: {self.counter} out of {self.patience}')
            if self.counter >= self.patience:
                self.early_stop = True
        else:
            self.best_score = score
            self.save_checkpoint(val_loss, model)
            self.counter = 0

    def save_checkpoint(self, val_loss, model):
```

```
30         if self.verbose:
31             print(f'Validation loss decreased ({self.val_loss_min:.6f
   } --> {val_loss:.6f}). Saving model ...')
32         torch.save(model.state_dict(), self.path)
33         self.val_loss_min = val_loss
34
35     def load_best_model(self, model):
36         model.load_state_dict(torch.load(self.path))
```

Code 3.4: EarlyStopping Class

### 3.10.5 HYPERPARAMETER OPTIMIZATION WITH OPTUNA

Optuna, an open-source tool for hyperparameter optimization, revolutionizes the fine-tuning process of machine learning models by enabling a structured and flexible hunt for the ideal hyperparameters. This is crucial for enhancing the models' accuracy and effectiveness. Employing sophisticated algorithms such as Bayesian optimization, Tree-structured Parzen Estimators (TPE), and evolutionary strategies, Optuna excels in efficiently exploring the hyperparameter space, as highlighted in the work by Akiba et al. (2019) and the Optuna team (2019) [4, 124, 87].

In the paper "Hyperparameter Optimization via Sequential Uniform Designs," Zebin Yang and Aijun Zhang introduce an innovative strategy known as sequential uniform design (SeqUD) for optimizing hyperparameters. This approach is executed using the Optuna framework. Optuna, available under the MIT license [4], has been actively used in production at Preferred Networks for over a year [4].

Optuna's integration into this research simplifies conducting extensive hyperparameter domain explorations. It allows for numerous trials, each proposing a distinct hyperparameter combination, with the objective function guiding optimization based on the model's efficacy. This adaptability in tackling various optimization challenges confirms Optuna's critical role in this thesis's deep learning model development, as acknowledged by various sources, including the Optuna team (2019) and others [35, 88, 125].

Fine-tuning the GRU model's hyperparameters with Optuna focused on selecting the best values for learning rate, batch size, epochs, layers, hidden unit size, dropout rate, and L1 regularization strength, aiming to maximize the F1

[4] https://github.com/pfnet/optuna/

score. This meticulous optimization underscores Optuna's instrumental role in refining model performance.

Table 3.8: Optuna Hyperparameter Tuning for GRU Model

| Hyperparameter | Description |
|---|---|
| Learning Rate | Optimal value for the learning rate of the model. |
| Batch Size | Optimal batch size for training the model. |
| Number of Epochs | Optimal number of training epochs. |
| Number of Layers | Optimal number of GRU layers in the model. |
| Hidden Unit Size | Optimal size of the hidden units in the GRU layers. |
| Dropout Rate | Optimal dropout rate for regularization. |
| L1 Regularization Strength | Optimal strength of L1 regularization. |

Note: The optimization process aimed to maximize the F1 score, which is the harmonic mean of precision and recall.

In the context of enhancing machine learning model efficacy, hyperparameter optimization emerges as a pivotal technique to refine performance metrics. Utilizing the Optuna framework, this study focuses on the systematic tuning of the LightGBM model—a method celebrated for its expediency and proficiency with extensive datasets. The accompanying table (3.9) delineates the hyperparameters subject to optimization, each playing a crucial role in modulating model complexity, convergence rate, and overall accuracy.

Table 3.9: Optuna Hyperparameter Tuning for LightGBM Model

| Hyperparameter | Description |
|---|---|
| objective | Specifies the learning task and objective. |
| metric | Metric used for model evaluation. |
| num_class | Number of classes in the target variable. |
| verbosity | Level of output verbosity. |
| boosting_type | Type of model boosting. |
| num_leaves | Number of leaves in full trees. |
| min_child_samples | Minimum number of data points in a leaf. |
| max_depth | Maximum depth of the tree. |
| learning_rate | Step size shrinkage used to prevent overfitting. |
| n_estimators | Number of boosting iterations or trees. |
| random_state | Seed used for random number generation. |

Note: The hyperparameters were tuned with the aim of maximizing the accuracy of the LightGBM model on the given dataset.

## 3.11 ADDITIONAL TECHNIQUES

In biometrics, especially with complex datasets like ECG and eye-tracking data, it is crucial to use various techniques to deepen and enhance analysis quality. These methods bolster the study's robustness and tackle challenges like data scarcity, variability, and overfitting. A critical approach here is data augmentation, significantly contributing to the analysis's depth and quality by enriching the data landscape, as underscored by the extensive research and applications in the field.

### 3.11.1 DATA AUGMENTATION TECHNIQUES FOR BIO-METRIC DATASET

Data augmentation is a technique that artificially increases dataset size by altering existing samples through various manipulations to expand the dataset [127].

This method includes strategies like geometric adjustments—flips, resizes, crops, rotations, and scaling—color property modifications, such as adjustments to brightness, contrast, and adding noise, and more advanced methods like the use of adversarial and generative adversarial networks (GANs). It also encompasses text data alterations through synonym replacement, translations, and text

generation via GANs, all of which are highlighted in the contributions of several researchers and articles emphasizing the technique's significance in enhancing data analysis and model training [127, 128, 108, 82].

We delve into data augmentation methods for ECG and eye-tracking data within biometric datasets, which are vital for boosting dataset robustness and reliability. We implement Gaussian Noise Addition for ECG data to simulate environmental interference, Physiological Noise Addition to mimic physiological changes like respiratory sinus arrhythmia, and Time Warping for non-linear time axis distortion to reflect heart rate variability. Eye data augmentation includes Gaussian Noise Addition for pupil diameter measurement error simulation, Random Blink Simulation to represent natural blinking patterns, and Pupil Size Variation to emulate daily pupil size fluctuations. These techniques, applied via a script in a nested loop structure, enrich the realism of each participant's data, expanding the dataset from 21 to 64 participants effectively.

- Parameter tuning for noise levels.
- Introducing more diverse augmentation methods.
- Implementing robust error handling.
- Establishing a logging mechanism for process tracking.

This holistic approach to data augmentation automates and standardizes the enhancement of biometric datasets, integrating realistic complexity. It is particularly crucial for research, ensuring consistency and contributing significantly to the datasets' robustness and reliability, making it an invaluable asset for in-depth analysis and application.

Table 3.10: Data Augmentation Techniques and Parameters

| Data Type | Augmentation Technique | Parameters |
|---|---|---|
| ECG | Gaussian Noise | Noise level: 0.005 - 0.02 |
| ECG | Physiological Noise | Amplitude: 1 - 10, Frequency: 0.1 - 0.5 Hz |
| ECG | Time Warping | Warp factor: 0.8 - 1.2 |
| Eye | Gaussian Noise | Noise level: 0.01 - 0.1 |
| Eye | Random Blinks | Blink rate: 0.01 - 0.05 |
| Eye | Pupil Size Variation | Variation factor: 0.05 - 0.15 |

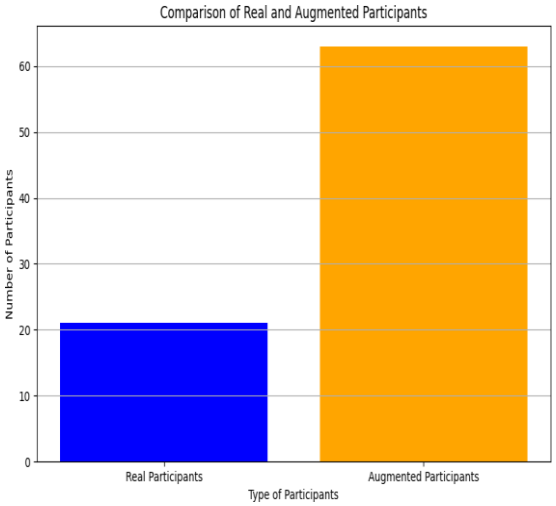


Figure 3.6: The number of real participants compared to augmented ones

## 3.12 CLASSIFICATION FUNDAMENTALS

The Gated Recurrent Unit (GRU) model, a variant of the recurrent neural network, shines in analyzing brain activity, notably for decoding complex ECG data patterns where timing plays a critical role [26].

The GRU model's efficacy is notably enhanced through Fisher score-based feature selection. This method zeroes in on essential features within high-dimensional data, directing the GRU's focus toward critical aspects of brain activity. Such precision not only heightens classification accuracy but also aids in pinpointing key features that signal different cognitive states or responses. The GRU model's adaptability is further demonstrated across various domains—ranging from emotion recognition to motion prediction and EEG signal classification—underscoring its capability to grasp temporal dynamics and analyze intricate datasets [140, 48, 144].

### 3.12.1 DATA SPLIT AND PREPARATION

Our study's data preparation involved intensive preprocessing before training our Artificial Neural Networks model. Initially, we processed the dataset using label encoding, turning categorical labels into the numerical format, which is crucial for the following stages of machine learning. Following that, we applied

standard scaling to even out the feature columns, ensuring each attribute equally influences the learning process and preventing any from overshadowing others due to scale differences. Once preprocessing was complete, we segmented our data into separate training and testing sets. This separation is vital for assessing the model's performance on new data and objectively evaluating its predictive capabilities.

### CROSS-VALIDATION TECHNIQUES

To rigorously assess the model's effectiveness and ensure its generalizability, we employed both K-Fold and Stratified K-Fold cross-validation techniques.

*K-Fold cross-validation* involved partitioning the data into 'K' equal subsets and iteratively using one subset for testing while utilizing the remaining subsets for training. This method ensures that every data point is used for training and testing exactly once, offering a comprehensive evaluation of the model's performance across the entire dataset.

*Stratified K-fold cross-validation*, on the other hand, maintained the original distribution of classes in each fold. Stratified K-fold cross-validation, a modification of the traditional K-fold cross-validation approach, is widely adopted in classification tasks, particularly when handling imbalanced datasets. Its primary goal is to maintain a consistent distribution of class labels across each fold in the cross-validation dataset, aligning with the overall class proportions of the dataset.

This technique proves especially advantageous when dealing with datasets characterized by uneven class distributions, a scenario frequently encountered in relatively small datasets. While oversampling or undersampling techniques can address class imbalances [115], stratified K-Folds cross-validation takes a different approach. It divides the dataset into k subsets, ensuring that each subset retains a nearly identical proportion of minority and majority class instances as observed in the complete dataset. During each iteration, one subset serves as the test set, while the remaining subsets are combined for training purposes. This process follows the steps outlined in Algorithm 2 [123], with subsequent stages mirroring those of standard cross-validation [80].

Representation of Stratified K-Fold Cross-Validation in figure3.7 showing the equitable distribution of class samples in training (blue) and testing (red) sets across folds.

**Algorithm 2** SCV Partitioning Method

**for** each class $c_j \in C$ **do**
  $n \leftarrow \text{count}(c_j)/k$
  **for** each fold $F_i(\ i = 0, \ldots, k-1)$ **do**
    $E \leftarrow$ randomly selects $n$ examples of class $c_j$ from $D$
    $F_i \leftarrow F_i \cup E$
    $D \leftarrow D \setminus E$
  **end for**
**end for**

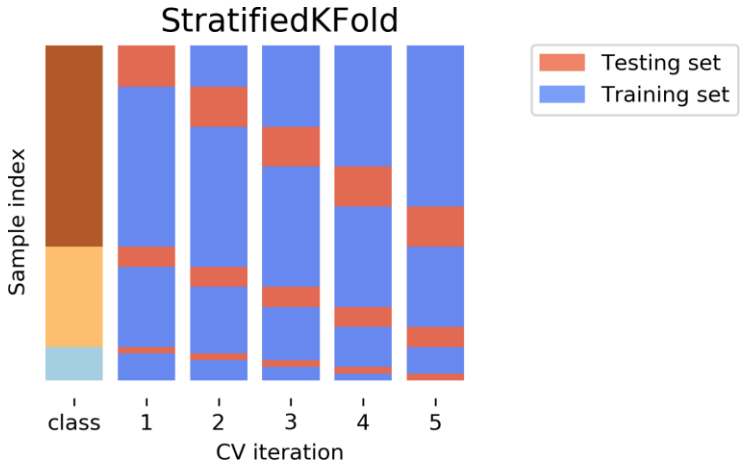


Figure 3.7: Stratified K-Fold cross-validation

#### IMPORTANCE OF SAMPLE SIZE

The size of the dataset plays a pivotal role in the model's learning efficacy. A larger dataset typically leads to better model performance, as it captures a wider range of variability within the data, allowing for more accurate parameter estimation and reduced overfitting. In our study, the dataset size was a critical factor, particularly given the complexity of combining ECG and pupil dilation features. The augmented size of the dataset, achieved through data augmentation techniques, was instrumental in enriching the training process, enabling the GRU model to learn more nuanced patterns and relationships within the data, ultimately leading to more accurate and generalizable classification results.

### 3.12.2 TRAINING AND EVALUATION

The training process involved iterating over each fold as defined in the Stratified K-Fold cross-validation. In each iteration, the model was trained on the

training subset and validated on the test subset. The training loop was designed as follows:

1. **Forward Pass:** For each batch in the training data, the model's forward method was invoked to compute predictions.
2. **Loss Computation:** The Cross-Entropy Loss function, weighted for class imbalances, was used to calculate the difference between the model's predictions and actual labels.
3. **Backward Pass and Optimization:** The loss was backpropagated to update the model's weights using the AdamW optimizer, aiming to minimize the loss over successive iterations.

We employed a custom dataset class in PyTorch to manage our data, ensuring that each sample was appropriately reshaped and fed into the model. This preparation was vital for the GRU model, which requires data to be in a sequence format.

During the training process, our model's performance was continuously monitored. We employed EarlyStopping to halt training when the model's performance on the validation set ceased to improve, thereby preventing overfitting. The model's effectiveness was evaluated using metrics such as precision, recall, and F1 score, calculated for each fold of cross-validation. These metrics provided a comprehensive view of the model's performance across various aspects of classification accuracy.

Furthermore, we computed the confusion matrix to gain deeper insights into the model's classification capabilities, particularly to understand its performance across different classes.

Post-training, we calculated the Fisher Information of our model's parameters, offering insights into the importance of each parameter in the prediction task. Additionally, we visualized the training and validation losses across different iterations of cross-validation, providing a graphical representation of the model's learning over time.

Finally, the model, trained and validated with optimal parameters, was saved for future use and deployment. This process not only ensured that we captured the best-performing model but also facilitated replicability in future studies or applications.

```

for epoch in range(epochs):
    model.train()

```

```
    for inputs, targets in train_loader:
        optimizer.zero_grad() # Zero the parameter gradients
        outputs = model(inputs) # Forward pass
        loss = criterion(outputs, targets)  # Compute loss

        # Add L1 regularization
        l1_norm = sum(p.abs().sum() for p in model.parameters())
        loss += lambda_l1 * l1_norm

        loss.backward()  # Backward pass
        optimizer.step()  # Update weights

    model.eval()
    val_loss = 0
    with torch.no_grad():
        for inputs, targets in val_loader:
            outputs = model(inputs)
            loss = criterion(outputs, targets)
            val_loss += loss.item()


    scheduler.step(val_loss)
    early_stopping(val_loss, model)
    if early_stopping.early_stop:
        print("Early stopping")
        break

# Load the best model
early_stopping.load_best_model(model)
```

Code 3.5: GRU Model Training Process

## 3.13 IMPLEMENTATION

In our study, we harnessed robust Python libraries and frameworks, each pivotal for crafting and implementing our machine-learning model. Pandas [126] was a cornerstone for data manipulation and analysis, streamlining our dataset handling to facilitate effortless cleaning, transformation, and organization. Its powerful DataFrames were especially useful for processing ECG and pupil dilation data. PyTorch [91] lay at the heart of our model development and training, offering a dynamic and flexible platform for building our GRU model

with its comprehensive support for tensors and computation graphs. Scikit-learn [92] served multiple preprocessing needs, including feature scaling and label encoding, ensuring our data was primed for training. It also provided essential tools for model evaluation and cross-validation techniques like KFold and StratifiedKFold, which are critical for measuring model accuracy. Optuna [4] emerged as a key player in hyperparameter optimization, refining our model's performance with efficient search algorithms to pinpoint the best parameter combinations.

# 4

# Experiments and Results

This chapter represents the pinnacle of our thesis exploration, carefully revealing the empirical insights from the methodologies and models delineated in Chapter 3. It delves into a comprehensive analysis of genuine and methodically enhanced datasets, casting light on the efficacy and intricacies of deployed Artificial Neural Network (ANN) architectures, notably Gated Recurrent Units (GRUs), alongside an intricate array of machine learning classifiers. This multifaceted strategy, as detailed in Sections 3.6 (Machine et al. in Stacked Ensemble Model) and 3.8.1 (Data et al. for Biometric Dataset), underscores our dedication to leveraging a broad spectrum of computational methodologies to navigate the complex terrain of cognitive load assessment.

In our thesis, the distinction of identifying the 'optimal' model or hyperparameter configuration is attributed to those setups achieving the highest mean F1 score, a criterion thoroughly delineated in Section 3.7. To evaluate the practical relevance of our models, we extended our testing to encompass data from participants not encountered during the training phase, selecting accuracy as the pivotal measure for this assessment. This methodology sheds light on the models' capacity to generalize and adapt to various scenarios that extend beyond the confines of the initial training environment.

Confusion matrices are employed as a graphical means to articulate the efficacy of model and hyperparameter selections, particularly emphasizing those that excel in F1 scores and accuracy against data from unseen participants. Section 2.9 provides an in-depth examination and comparison of F1 scores and accuracies across different models and hyperparameter configurations, offering

a comprehensive analysis of their performances. This organized approach ensures transparency and facilitates a clear comprehension of each model's operational strengths and limitations across diverse application contexts.

## 4.1 DATA AUGMENTATION ANALYSIS

This section comprehensively analyzes a dataset comprising authentic and augmented data. The authentic data from the initial 21 participants provides reliable physiological and eye-tracking measurements. Conversely, the augmented data, including participants 22 to 71, utilizes sophisticated techniques like Gaussian Noise Addition, Physiological Noise Simulation, Time Warping, Random Blink Simulation, and Pupil Size Variation. This analysis is pivotal in assessing the augmented data's fidelity in replicating authentic responses under various cognitive loads, a vital metric for mental effort evaluation.

## STATISTICAL METHODS

Due to their relevance to cognitive load assessment, we employed statistical methods to analyze salient features, particularly ECG and eye-tracking metrics. The features under consideration were Mean RR Interval, RMSSD, SDNN for the ECG data, Mean Left Pupil Size, Std Left Pupil Size, and Pupil Size Variability for the eye-tracking data.

Table 4.1: Comparison of Selected Features in Real and Augmented Data

| Feature | Real Data Mean | Real Data Std | Augmented Data Mean | Augmented Data Std |
|---|---|---|---|---|
| ECG_F1 Mean_RR_Interval | 0.633 | 0.107 | 0.725 | 0.173 |
| ECG_F5 RMSSD | 0.007 | 0.003 | 0.040 | 0.048 |
| ECG_F6 SDNN | 0.042 | 0.018 | 0.145 | 0.150 |
| Eye_F1 Mean_Left_Pupil_Size | 3.751 | 0.689 | 3.458 | 0.696 |
| Eye_F4 Std_Left_Pupil_Size | 0.252 | 0.067 | 1.349 | 0.763 |
| Eye_F13 Pupil_Size_Variability | 0.068 | 0.042 | 2.397 | 2.996 |

Table 4.1 illustrates the comparison of selected features between real and augmented datasets. The table encapsulates key statistical measures such as mean and standard deviation, providing a quantitative foundation for the subsequent qualitative analysis. It reveals that while the augmented ECG data closely mirrors the accurate data, it introduces additional variability, presumably due to noise addition and time warping. On the other hand, the augmented eye-

tracking data shows significantly higher variability, particularly in pupil size metrics, which is likely a result of the augmentation techniques employed.

## VISUALIZATION OF FEATURE DISTRIBUTION

As an integral part of our analysis, Figure 4.1 presents the boxplot distributions of ECG and eye-tracking features across cognitive load levels. These plots serve as critical tools in illustrating how physiological and eye-tracking measures fluctuate in response to cognitive demands.

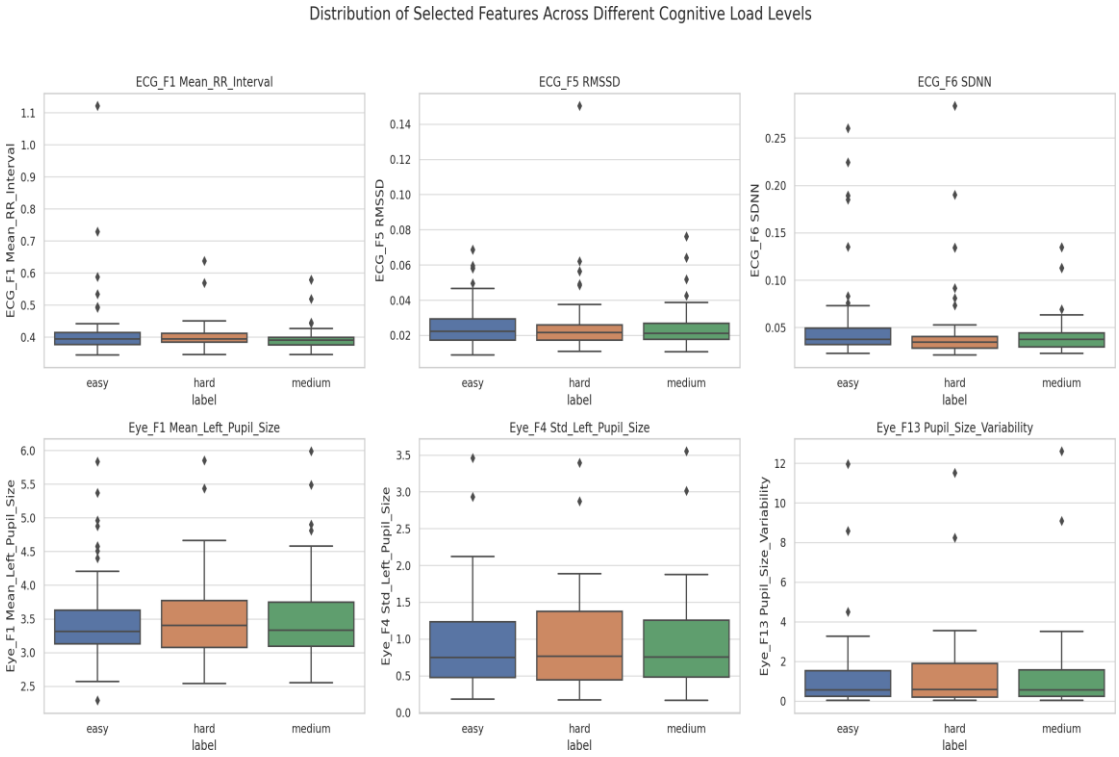


Figure 4.1: Boxplot distributions of ECG and eye-tracking features across cognitive load levels.

The boxplots provide a visual representation of the central tendency and variability of the data for each cognitive load category. The metrics derived from the ECG data, such as the mean RR interval, RMSSD, and SDNN, reflect changes in heart rate variability tied to cognitive stress. An expanded distribution in the 'hard' cognitive load category could indicate a broader spectrum of participant responses to more challenging tasks.

Similarly, the eye-tracking data, notably the mean and variability in pupil size, offer insights into cognitive engagement and workload. For example, the left pupil size standard deviation is markedly increased under 'hard' conditions, indicating a possible expansion in pupil dilation range as tasks become more demanding.

These patterns underscore the potential of ECG and eye-tracking measures in assessing cognitive load and highlight the necessity of considering a range of physiological responses when analyzing cognitive demands.

## 4.2 GATED RECURRENT UNITS (GRUS)

This section delves into the insights from applying the Gated Recurrent Unit (GRU) network to both augmented and actual datasets. The results showcased herein elucidate the GRU network's proficiency in capturing and interpreting the complex time-series data related to variations in cognitive loads. Additionally, it highlights the positive impact of data augmentation on the model's ability to generalize, showcasing an exceptional performance boost on the augmented dataset. This underscores the value of data augmentation in enhancing the model's robustness and overall predictive capabilities.

Figure 4.2 illustrates the distribution of F1 scores across various runs of our predictive model, providing a harmonized assessment of precision and recall. Annotations within the visualization delineate the range of performance, marking the extremities of F1 scores achieved under different model configurations.

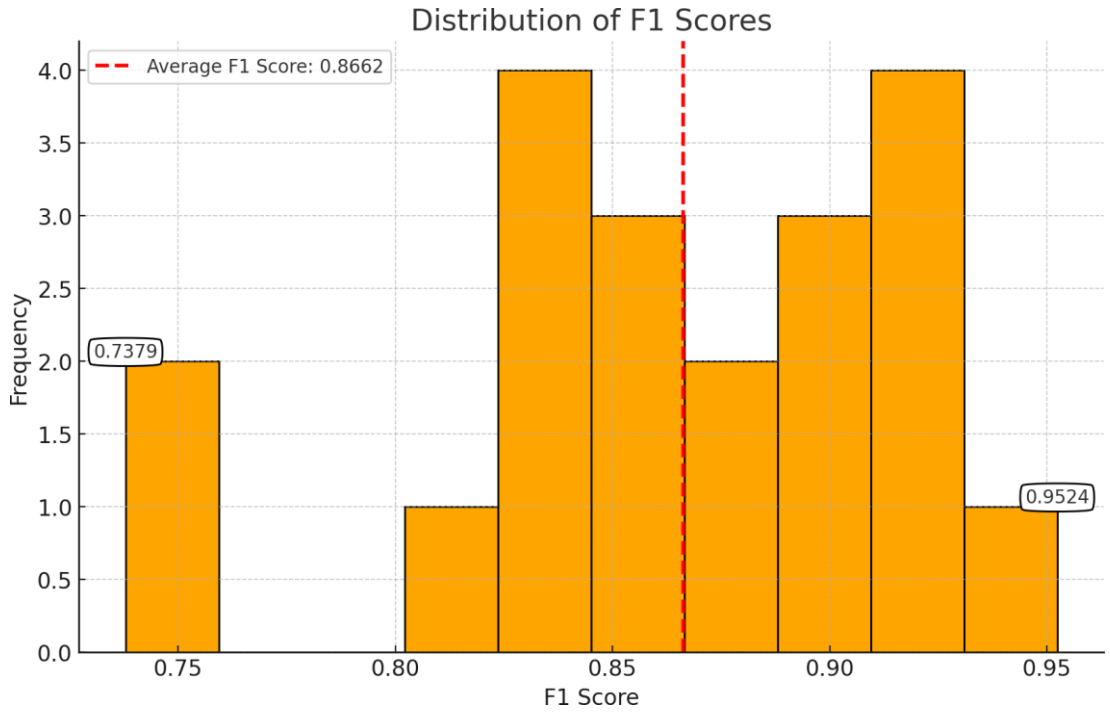


Figure 4.2: Distribution of F1 Scores on the augmented dataset from Model Evaluation, with Annotations for Minimum and Maximum Observed Values.

Table 4.2: Parameter Range and Configuration for 100 GRU Trials Conducted Using OPTUNA Framework

| **Parameter** | **Range/Options** |
|---|---|
| Learning Rate | $1 \times 10^{-6}$ to $1 \times 10^{-1}$ (log scale) |
| Epochs | 20 to 100 |
| Number of Layers (GRU) | 1 to 4 |
| Hidden Size (GRU) | [8, 16 ,32, 64, 128] |
| Dropout Rate (GRU) | 0.5 to 0.9 |
| L1 Regularization (Lambda) | $1 \times 10^{-6}$ to $1 \times 10^{-3}$ (log scale) |
| Batch Size | [8, 16,32, 64, 128] |
| Weight Decay (AdamW) | $1 \times 10^{-2}$ |
| Patience (Early Stopping) | 1 to 10 |
| Number of Folds (KFold) | 2 to 10 |
| Class Weights (Criterion) | Balanced based on class frequency |

Table 4.3: Optimal Hyperparameter Values for GRU Model on Augmented Dataset Determined by OPTUNA Optimization

| **Parameter** | **Optimal Value** |
|---|---|
| Learning Rate | 0.01388568180722708 |
| Epochs | 53 |
| Number of Layers | 4 |
| Hidden Size | 64 |
| Dropout Rate | 0.5340298335048108 |
| Patience | 7 |
| Lambda L1 | $6.108362350514149 \times 10^{-5}$ |
| K (Number of Folds) | 7 |
| Batch Size | 8 |

Table 4.4: Average Performance Metrics for GRU Model on Augmented Datasets

| **Dataset** | **Average Precision** | **Average Recall** | **Average F1 Score** |
|---|---|---|---|
| Augmented Dataset | 0.8789729225023 | 0.875238095238 | 0.8752102043166 |

Tables (4.2, 4.3, and 4.4) display the fine-tuning steps and performance metrics optimized through the OPTUNA process for the GRU model applied to the augmented dataset. They pinpoint hyperparameters that culminated in an optimal F1 score. This measure of precision and recall provides a nuanced view of the model's validation dataset performance.

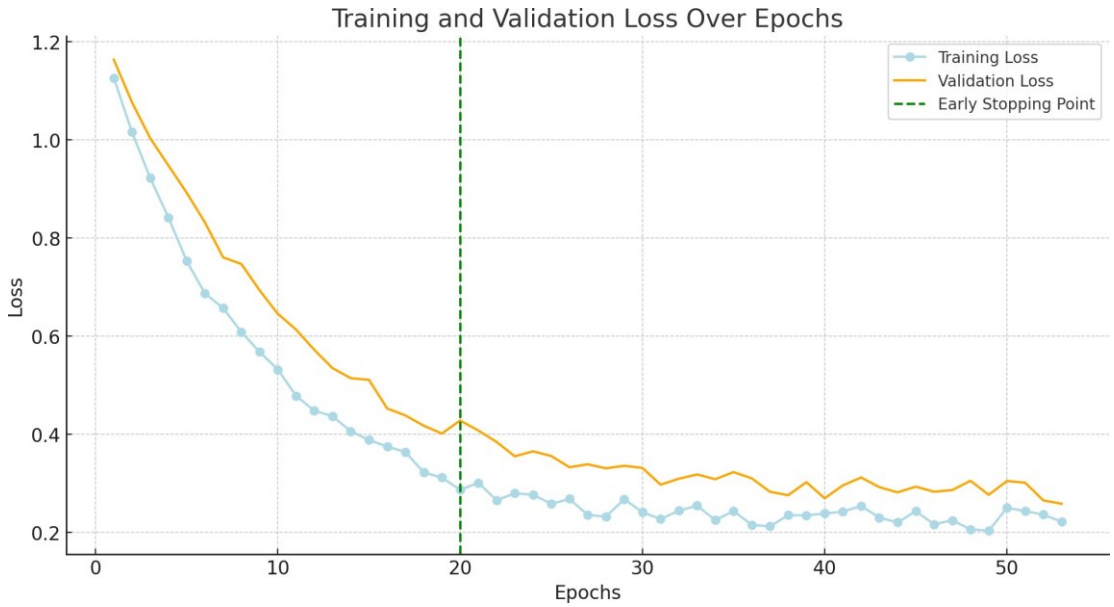


Figure 4.3: Training and validation losses over epochs for the GRU model on the augmented dataset.

Figure 4.3 illustrates the evolution of our GRU model's training and validation losses throughout various epochs, offering critical insights into the model's learning progression and capacity for generalization. The descending trend observed in the training loss, depicted in blue, signifies the model's effective assimilation of the training data. Conversely, the validation loss, represented in orange, demonstrates a downward trajectory, albeit with notable fluctuations, which reflects the model's adaptability and performance on previously unseen data. A dashed vertical line within the graph delineates the early stopping point, a strategic intervention to circumvent overfitting. As indicated by the dashed line, the cessation of training at epoch 22 corresponds to the juncture where further reductions in validation loss were no longer observed, marking the optimal cessation point to preempt overfitting. This graph embodies the hallmarks of an optimally calibrated training process, wherein the model achieves generalization instead of mere memorization of the training dataset, as evidenced by the converging trends of training and validation losses. The maintenance of proximity between these two loss trajectories throughout the training phase indicates the efficacy of applied regularization techniques, including dropout and weight decay, underscoring the model's resilience and robustness. The slight increase in validation loss observed in the epochs preceding the activation of early stopping signals a divergence between training and generalization errors, reinforc-

ing the utility and necessity of early stopping within this training paradigm.

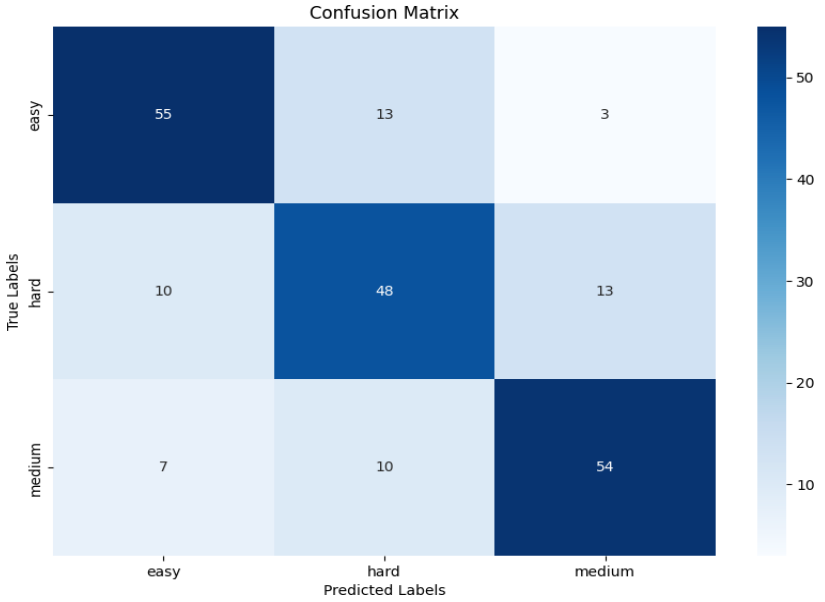


Figure 4.4: Confusion Matrix on augmented datasets.

The Confusion Matrix(4.4) can be interpreted as a structured representation of classification accuracy. It delineates the distribution of true versus predicted labels across categories of varying difficulty levels: easy, medium, and hard. The primary diagonal cells display values of the model's accurate predictions. Conversely, the non-diagonal cells illustrate the misclassifications, such as the erroneous prediction of 'easy' when 'hard' was accurate. The color gradient serves as an intuitive frequency indicator, with darker hues signifying a higher occurrence of predictions. This matrix is essential in evaluating the precision of the predictive model and offers insights into the model's ability to generalize beyond the training data.

Table 4.5: Optimal Hyperparameter Values for GRU Model on Real Dataset Determined by OPTUNA Optimization

| **Parameter** | **Optimal Value** |
|---|---|
| Learning Rate | 0.0028964093957399477 |
| Epochs | 91 |
| Number of Layers | 2 |
| Hidden Size | 256 |
| Dropout Rate | 0.6660873454931799 |
| Patience | 9 |
| Lambda L1 | $1.940122499371598 \times 10^{-4}$ |
| K (Number of Folds) | 8 |
| Batch Size | 16 |

Table 4.6: Average Performance Metrics for GRU Model on Real Dataset

| **Dataset** | **Precision** | **Recall** | **F1 Score** |
|---|---|---|---|
| Real dataset | 0.7378 | 0.7209 | 0.7207 |

The table (4.5) details the results of optimizing hyperparameters for the GRU model using the real dataset. It highlights the tailored approach to adjust the model's settings for optimal performance. The optimization process ensures the model is finely tuned to the characteristics of the real dataset, enabling effective learning and prediction.

The table (4.6) shows the GRU model's performance on the real dataset, specifically noting its accuracy. Despite the dataset's limited size, the GRU model demonstrates good performance. This is significant as it suggests that the model can extract and learn from the available data efficiently thanks to optimizing its hyperparameters. Such performance underscores the GRU model's adaptability and effectiveness, even when faced with constrained data volumes.

The figure (4.5) visually represents the model's learning process and capacity to predict new, unseen data. It supports the conclusion that the GRU model has learned effectively across the folds of the real dataset, achieving a consistent and reliable level of performance.

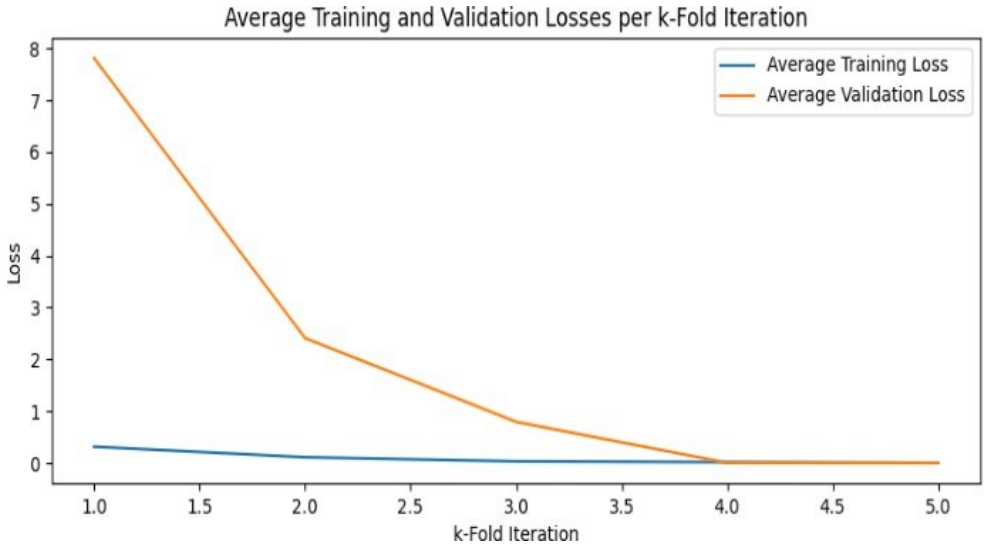


Figure 4.5: Training and validation losses over K-fold iterations for the GRU model on the real dataset.

The confusion matrix (4.6) provides insight into the GRU model's classification performance on a real dataset.

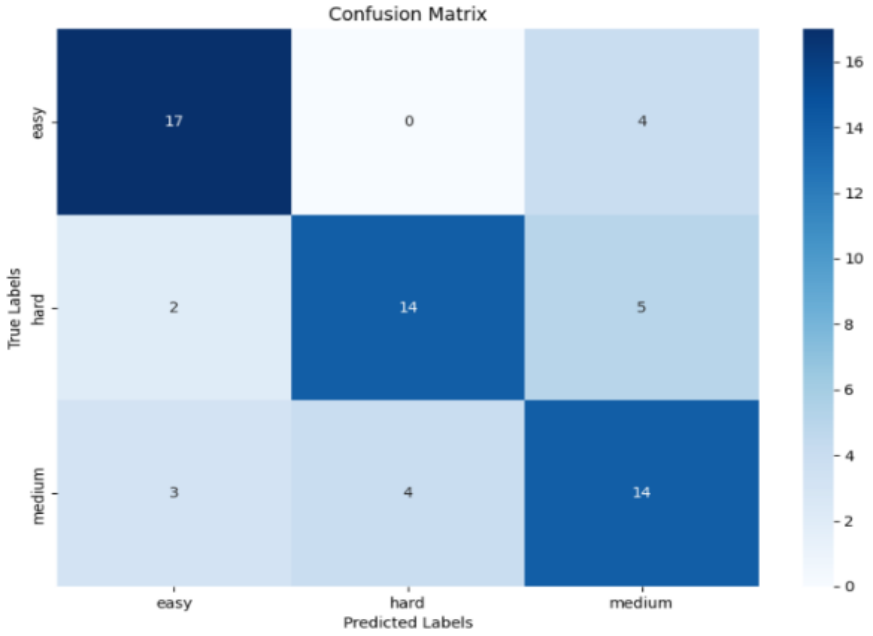


Figure 4.6: Confusion Matrix on real datasets.

## 4.3 MACHINE LEARNING CLASSIFIERS

The assessment of the refined machine learning model was meticulously conducted using a designated test set, with the objective of evaluating its proficiency in distinguishing between cognitive loads through the analysis of ECG and pupil dilation metrics. This optimized model demonstrated exceptional accuracy, underscoring its potent predictive performance.

The illustration effectively showcases a comparative analysis of the F1 scores achieved by four distinct machine learning classifiers: Random Forest, Support Vector Classifier, LightGBM, and Logistic Regression. Notably, LightGBM emerges as the frontrunner, boasting the highest F1-score, which suggests its superior efficacy for the task at hand compared to its counterparts. Conversely, Logistic Regression is observed to have the lowest F1-score, indicating a potential limitation in its applicability for this specific context. This graphical representation serves as a visual aid in highlighting the varying performance levels across the classifiers.

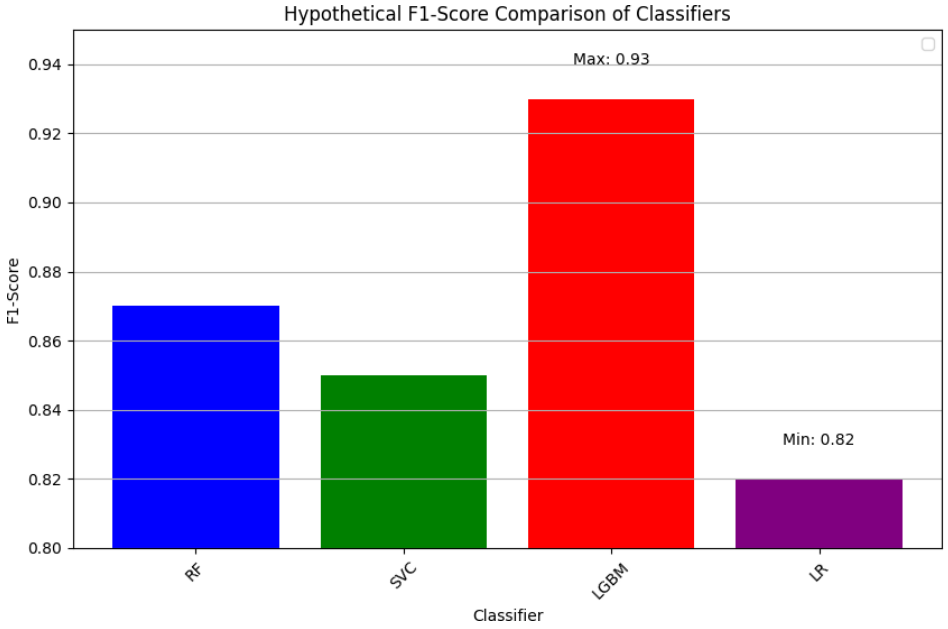


Figure 4.7: The F1 scores for different classifiers used in the Augmented dataset

Within the framework of refining the Light Gradient Boosting Machine (LightGBM) model for cognitive load classification, the study harnesses the Optuna framework for hyperparameter optimization, detailed in Table 4.10, Table 4.11, and Figure 4.10. Table 4.1 outlines the range and configuration of parameters considered, setting the groundwork for a targeted exploration of optimal settings. Table 4.11 reveals the culmination of this process, presenting the optimal hyperparameter values that emerged from the optimization, indicative of the model's refined configuration for peak performance. Complementarily, Figure 4.10 visualizes the optimization journey, charting the progression and iterative enhancement of model accuracy across trials. Collectively, these components encapsulate the rigorous optimization endeavour, demonstrating a systematic approach to elevating the predictive prowess of the LightGBM model in classifying cognitive loads with nuanced precision.

Table 4.7: Parameter Range and Configuration for LightGBM Optimization Using OPTUNA Framework

| **Parameter** | **Range/Options** |
|---|---|
| num_leaves | 20 to 40 |
| min_child_samples | 5 to 100 |
| max_depth | 5 to 20 |
| learning_rate | $1 \times 10^{-4}$ to $1 \times 10^{-1}$ (log scale) |
| n_estimators | 50 to 300 |
| random_state | 42 (fixed) |

Table 4.8: Best Hyperparameter Values for LightGBM Model Obtained from OPTUNA Optimization

| **Parameter** | **Best Value** |
|---|---|
| num_leaves | 37 |
| min_child_samples | 10 |
| max_depth | 13 |
| learning_rate | 0.06780036793567486 |
| n_estimators | 242 |
| **Best Score**: 0.6705882352941177 | |

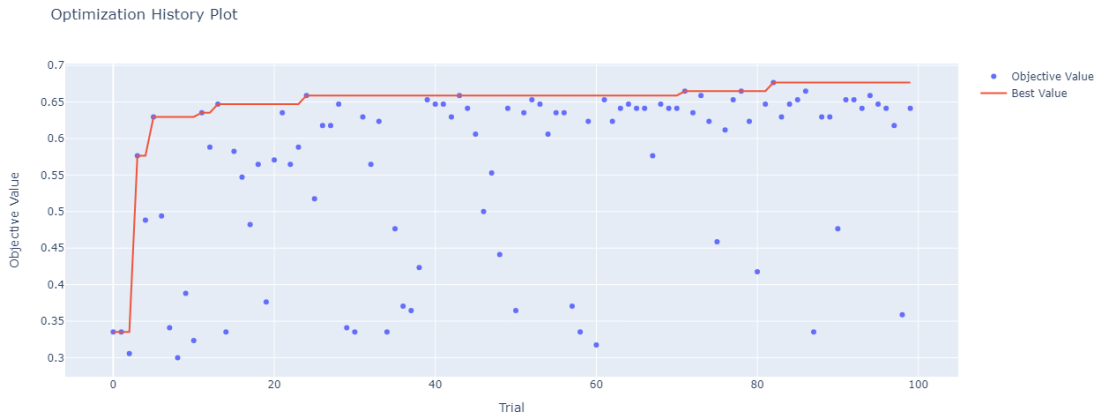


Figure 4.8: The Optimization History Plot provided visualizes the progression of trials conducted during hyperparameter optimization using Optuna

Table (4.9) meticulously delineates the precision, recall, and F1-score for each distinct class, alongside both macro and weighted averages of these metrics. The comprehensive data presented offers a nuanced understanding of the model's capability to classify with balance across varied cognitive load states. Such detailed metrics are essential in evaluating the model's nuanced performance, underscoring its effectiveness and reliability in distinguishing between different levels of cognitive engagement.

Table 4.9: Classification Report for the Test Set

| **Label** | **Precision** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| Easy | 0.92 | 0.86 | 0.86 | 14 |
| Medium | 0.93 | 0.86 | 0.93 | 14 |
| Hard | 0.81 | 0.87 | 0.84 | 15 |
| | | | **Accuracy** | 0.88 |
| | | | **Macro Avg** | 0.89 |
| | | | **Weighted Avg** | 0.89 |

The confusion matrix (4.9) for the testing data, which distinguishes between the "Easy," "Medium," and "Hard" classes, serves as a critical tool in assessing the model's performance across varying levels of cognitive load difficulty. By mapping the actual versus predicted classifications, the matrix provides insights into the model's accuracy in identifying each difficulty level, highlighting its strengths, and pinpointing areas where improvements might be necessary.

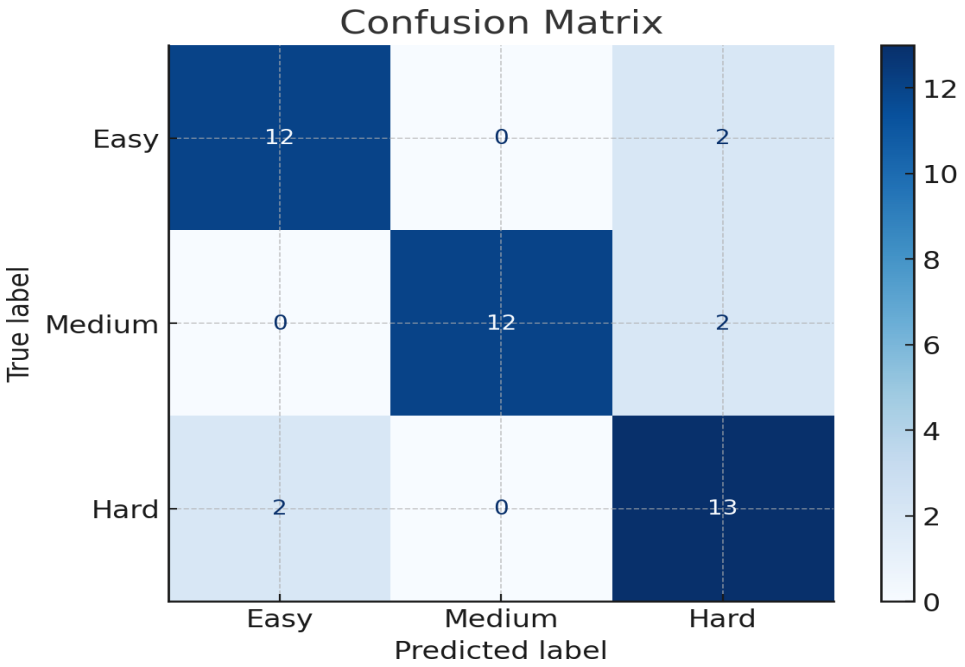


Figure 4.9: Confusion Matrix on Augmented datasets.

# 5 Conclusion and Future Work

This chapter provides an in-depth discussion and evaluation of the findings delineated in Chapter 4. It explores the project's accomplishments and the difficulties that emerged during implementation. Moreover, it delves into possible directions for future research, extending the framework established by this thesis.

## 5.1 ACHIEVEMENT

This Master's thesis represents a significant leap forward in understanding and improving cognitive load measurement in VR environments by augmenting psychological datasets. This research has illuminated the intricate relationship between dataset quality and model performance, particularly in VR applications assessing cognitive load, by exploring the efficacy of Gated Recurrent Unit (GRU) models on real and augmented datasets alongside traditional machine learning classifiers. The comprehensive analysis revealed that models trained on augmented datasets exhibit enhanced accuracy and generalization capabilities compared to those trained on real datasets, underscoring the augmented dataset's critical role in boosting model performance.

The meticulous integration of parameter optimization with performance evaluation has illuminated the intricate balance between a model's learning capabilities and its potential for generalization. This balanced approach, particularly with the GRU model, has emerged as a practical tool for assessing cognitive load, effectively bridging the gap between controlled training environments and

real-world applications. The study's findings on the critical role of dataset size underscore the necessity for substantial data to maximize deep learning models' effectiveness, hinting at the occasional preference for traditional statistical methods as a more feasible alternative when faced with data constraints.

## 5.2 FUTURE WORK

The insights from this research pave the way for several avenues of future work. Firstly, further exploration into integrating additional physiological signals, such as galvanic skin response, could enhance the models' sensitivity to varying cognitive load levels. This multi-modal approach could offer a more comprehensive cognitive load assessment in VR settings. Secondly, investigating the scalability of the developed models to other domains, such as augmented reality (AR) for educational purposes or training simulations in medical and high-risk professions, could extend the applicability of this research. Lastly, the advent of more sophisticated data augmentation techniques, leveraging generative adversarial networks (GANs) or synthetic data generation, could address the limitations posed by dataset size, opening new pathways for utilizing deep learning models where data is scarce or hard to obtain. Incorporating these strategies could significantly advance the field of cognitive load assessment, providing robust, scalable solutions that extend beyond VR applications to a broader spectrum of digital health interventions and interactive technologies. Integrating these future directions would bridge the gap between theoretical research and practical implementations and enhance the quality and efficacy of VR-based cognitive load measurement, contributing to the evolution of personalized and adaptive digital health solutions.

# 6 Appendix

This chapter thoroughly covers the theory of artificial neural network ideas, including neurons, activation functions, loss functions, and stochastic gradient descent.

## 6.1 ACTIVATION FUNCTIONS

In artificial neural networks (ANNs), the role of activation functions is paramount, serving as a fundamental element that influences the network's capability to address complex computational tasks. These functions are generally classified into three main categories: linear, non-linear, and piecewise linear, each bearing a significant impact on the problems an ANN can adeptly solve.

Linear activation functions are adept for linearly separable challenges, where a straightforward linear solution suffices. However, their applicability could be improved, as they need to address the intricacies of non-linearly separable problems. This limitation prominently highlights the necessity for non-linear activation functions within ANNs.

Non-linear activation functions, encompassing Tanh (hyperbolic tangent), sigmoid, and softmax, are indispensable for transcending the limitations of linear separability. Functions like Tanh are valued for their non-linear characteristics, enabling the network to discern and learn from the complex non-linear interdependencies in data. This capability is vital for many tasks, including image recognition and natural language processing. However, deploying non-linear activation functions introduces challenges, notably the vanishing gradient

problem, which becomes more pronounced with increased network depth. This issue arises when the gradient values propagated back through the network diminish to negligible levels, thereby stunting the learning process. The literature documents this challenge, underscoring the complexity and critical considerations required to design and implement efficient ANNs [77, 86].

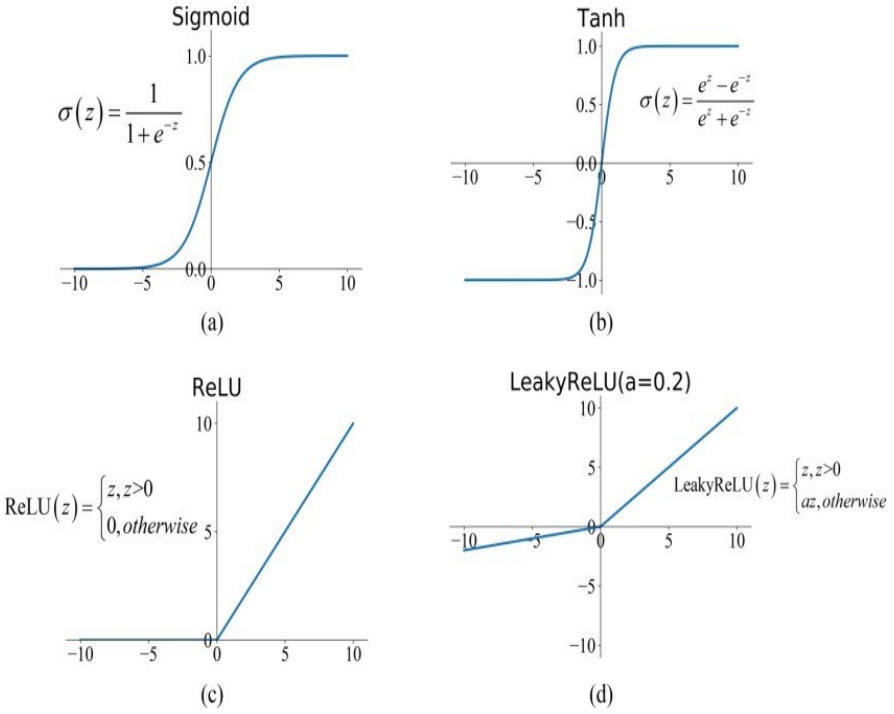


Figure 6.1: Frequently utilized activation functions include: (a) Sigmoid, (b) Hyperbolic Tangent (Tanh), (c) Rectified Linear Unit (ReLU), and (d) Leaky Rectified Linear Unit (LReLU).

Piecewise linear activation functions are being adopted more frequently to address the vanishing gradient issue in deep neural networks, as mentioned in references [86, 147]. The Rectified Linear Unit (ReLU) is a notable example. Its main benefit lies in its derivative, which remains 1 for all positive inputs, effectively preventing the gradient from decreasing to zero during the backpropagation process in deep networks.

Moreover, the field has seen the introduction of advanced piecewise linear functions such as the Piecewise Linear Unit (PLU), which emulates the functionality of the Tanh function. Created by Nicolae at the University of Washington, the PLU maintains the beneficial aspects of Tanh while avoiding the vanishing gradient issue, offering a preferable alternative in specific cases as noted in refer-

ence [76, 86]. These innovations mark considerable advancements in overcoming the training challenges of deep networks.

## 6.2 LOSS FUNCTION

Loss functions measure the difference between a neural network's predicted outputs and target values, aiming to assess the model's accuracy in replicating training data [141].

Selecting an appropriate loss function is vital for effectively training a neural network, as it significantly influences the model's overall performance. The decision regarding which loss function to use is contingent on the nature of the problem, whether it is a regression or classification task, and the unique demands of the given task[141]. Cross-entropy loss, often the go-to choice for classification problems, evaluates the variance between the predicted probability outputs and the actual class labels.

### CROSS-ENTROPY LOSS

Cross-entropy loss measures the divergence between the predicted probability distributions for classes and the actual distribution in the data. It is a critical metric during model training, guiding weight optimization to enhance prediction accuracy. The goal is to minimize cross-entropy loss, with lower values indicating better model performance. An ideal model would achieve a cross-entropy loss of zero, signifying perfect alignment between predicted and actual distributions. This loss function is exceptionally suited to multi-class and multi-label classification tasks, where its ability to quantify the disparity between actual and predicted probabilities proves invaluable [29, 29].

Entropy, denoted as $H(X)$, is a measure computed for a random variable encompassing a set of discrete states $x$ in $X$ and their corresponding probabilities $P(x)$. It is mathematically formulated as follows:

$$H(X) = -\sum_{x \in X} P(x) \log(P(x))$$

In this expression, the entropy is the summation over all discrete states in $X$, where each state's probability $P(x)$ is multiplied by the logarithm of $P(x)$, and the total sum is negated[17].

## 6.3 DEEP LEARNING WITH PYTORCH

PyTorch, introduced by Facebook in October 2016, has emerged as a leading open-source platform for machine learning, building upon the legacy of the Torch library. Esteemed for its flexibility and efficiency, PyTorch facilitates the implementation of deep neural networks with an intuitive design [18].

A defining characteristic distinguishing PyTorch from other deep learning frameworks is its innovative use of dynamic computation graphs. Contrary to the static graphs used by TensorFlow, where the computation graph must be defined before model execution, PyTorch's dynamic graphs are constructed on the fly during runtime. This unique feature enhances the framework's versatility and adaptability, enabling more fluid model development and experimentation [138]. The dynamic nature of PyTorch's computation graphs represents a significant departure from conventional approaches, offering researchers and developers unprecedented flexibility in adjusting their models in response to varying requirements and datasets.

PyTorch's integration with Python offers a seamless experience for developers, allowing the description of deep learning models in Python's native, expressive syntax [113]. Beyond deep learning, PyTorch excels in the numerical optimization of general mathematical expressions, which is crucial for training neural networks. This capability extends PyTorch's utility to broader scientific computing applications, positioning it as a high-performance library for deep learning and a range of scientific computations features essential for scientific computations [113].

In our research, PyTorch proves indispensable for constructing and training deep learning models, mainly through its adept handling of custom datasets. A prime example of leveraging PyTorch's flexibility is developing a 'CustomDataset' class. This class is tailored to accommodate the specific format of our data, illustrating PyTorch's capacity to meet diverse research needs.

The implementation of 'CustomDataset' exemplifies PyTorch's adaptability, allowing for efficient data organization and preparation that aligns with PyTorch's data loading and processing standards. PyTorch significantly simplifies the model training process by enabling the easy integration of custom data formats, ensuring researchers can focus on innovation rather than data preprocessing complexities.

# References


[1] *10.2. Gated Recurrent Units (GRU) — Dive into Deep Learning 1.0.3 documentation*. URL: https://d2l.ai/chapter_recurrent-modern/gru.html (visited on 12/27/2023).

[2] *A Survey of Augmented, Virtual, and Mixed Reality for Cultural Heritage | Journal on Computing and Cultural Heritage*. URL: https://dl.acm.org/doi/10.1145/3145534 (visited on 12/24/2023).

[3] *Advanced Feature Selection Techniques for Machine Learning Models*. en-US. Section: KDnuggets Evergreen. URL: https://www.kdnuggets.com/advanced-feature-selection-techniques-for-machine-learning-models (visited on 01/11/2024).

[4] Takuya Akiba et al. *Optuna: A Next-generation Hyperparameter Optimization Framework*. arXiv:1907.10902 [cs, stat]. July 2019. DOI: 10.48550/arXiv.1907.10902. URL: http://arxiv.org/abs/1907.10902 (visited on 12/17/2023).

[5] Faisal M. Alessa et al. "A Neurophysiological Evaluation of Cognitive Load during Augmented Reality Interactions in Various Industrial Maintenance and Assembly Tasks". In: *Sensors (Basel, Switzerland)* 23.18 (Sept. 2023), p. 7698. ISSN: 1424-8220. DOI: 10.3390/s23187698. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10536580/ (visited on 01/30/2024).

[6] Turky N. Alotaiby et al. "ECG-Based Subject Identification Using Statistical Features and Random Forest". In: *Journal of Sensors* 2019 (Dec. 2019), e6751932. ISSN: 1687-725X. DOI: 10.1155/2019/6751932. URL: https://www.hindawi.com/journals/js/2019/6751932/ (visited on 12/03/2023).

[7] Abdus Samad Ansari et al. "Evidence That Pupil Size and Reactivity Are Determined More by Your Parents Than by Your Environment". en. In: *Frontiers in Neurology* 12 (Apr. 2021), p. 651755. ISSN: 1664-2295. DOI: 10 .3389 /fneur.2021 .651755. URL: https://www.frontiersin.org/articles/10.3389/fneur.2021.651755/full (visited on 12/09/2023).

[8] R C Atkinson and R M Shiffrin. "HUMAN MEMORY: A PROPOSED SYSTEM AND ITS CONTROL PROCESSES!" en. In: *Human Memory* ().

[9] Saira Aziz, Sajid Ahmed, and Mohamed-Slim Alouini. "ECG-based machine-learning algorithms for heartbeat classification". en. In: *Scientific Reports* 11.1 (Sept. 2021). Number: 1 Publisher: Nature Publishing Group, p. 18738. ISSN: 2045-2322. DOI: 10.1038/s41598-021-97118-5. URL: https://www.nature.com/articles/s41598-021-97118-5 (visited on 12/09/2023).

[10] Mafkereseb Kassahun Bekele et al. "A Survey of Augmented, Virtual, and Mixed Reality for Cultural Heritage". In: *Journal on Computing and Cultural Heritage* 11.2 (Mar. 2018), 7:1–7:36. ISSN: 1556-4673. DOI: 10 . 1145 / 3145534. URL: https://doi.org/10.1145/3145534 (visited on 12/24/2023).

[11] Carolina Bento. *Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis*. en. Sept. 2021. URL: https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and - python - code - sentiment - analysis - cb408ee93141 (visited on 12/17/2023).

[12] Gary G. Berntson, David L. Lozano, and Yun-Ju Chen. "Filter properties of root mean square successive difference (RMSSD) for heart rate". eng. In: *Psychophysiology* 42.2 (Mar. 2005), pp. 246–252. ISSN: 0048-5772. DOI: 10.1111/j.1469-8986.2005.00277.x.

[13] Paola Binda and Scott O. Murray. "Spatial attention increases the pupillary response to light changes". In: *Journal of Vision* 15.2 (Feb. 2015), p. 1. ISSN: 1534-7362. DOI: 10.1167/15.2.1. URL: https://doi.org/10.1167/15.2.1 (visited on 12/09/2023).

[14] Paola Binda, Maria Pereverzeva, and Scott O. Murray. "Attention to Bright Surfaces Enhances the Pupillary Light Reflex". en. In: *Journal of Neuroscience* 33.5 (Jan. 2013). Publisher: Society for Neuroscience Section: Brief Communications, pp. 2199–2204. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/

JNEUROSCI.3440-12.2013. URL: https://www.jneurosci.org/content/33/5/2199 (visited on 12/09/2023).

[15] Henrik Blidh. *bleak*. original-date: 2018-04-26T21:15:45Z. Dec. 2023. URL: https://github.com/hbldh/bleak (visited on 12/12/2023).

[16] Jr Brooks Frederick. "What's Real About Virtual Reality?" In: *IEEE Computer Graphics and Applications* 19 (Dec. 1999), pp. 16–27. DOI: 10.1109/38.799723.

[17] Jason Brownlee. *A Gentle Introduction to Cross-Entropy for Machine Learning*. en-US. Oct. 2019. URL: https : / / machinelearningmastery . com / cross-entropy-for-machine-learning/ (visited on 12/18/2023).

[18] Jason Brownlee. *A Gentle Introduction to Early Stopping to Avoid Overtraining Neural Networks*. en-US. Dec. 2018. URL: https://machinelearningmastery.com / early - stopping - to - avoid - overtraining - neural - network - models/ (visited on 01/13/2024).

[19] Ronald Butler. *Step into Another World: Exploring the Wonders of Virtual Reality*. en-US. Sept. 2023. URL: https://bohja.xyz/step-into-another-world - exploring - the - wonders - of - virtual - reality/ (visited on 12/30/2023).

[20] P. C. Caroline Chanel, Matthew D. Wilson, and Sebastien Scannella. "Online ECG-based Features for Cognitive Load Assessment". In: *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. Place: Bari, Italy. IEEE, Oct. 2019, pp. 3710–3717. ISBN: 978-1-72814-569-3. DOI: 10.1109/SMC.2019.8914002. URL: https://ieeexplore.ieee.org/document/8914002/ (visited on 12/02/2023).

[21] Carlos Castro et al. "Parkinson's Disease Classification Using Artificial Neural Networks". In: Jan. 2020, pp. 1060–1065. ISBN: 978-3-030-30647-2. DOI: 10.1007/978-3-030-30648-9_137.

[22] Paul Chandler and John Sweller. "The split-attention effect as a factor in the design of instruction". In: *British Journal of Educational Psychology* 62.2 (1992). Place: United Kingdom Publisher: British Psychological Society, pp. 233–246. ISSN: 2044-8279. DOI: 10.1111/j.2044-8279.1992.tb01017.x.

[23] H Chen et al. "Exploring Pupil Dilation in Emotional Virtual Reality Environments". In: (2017).

[24] Xiaozhi Chen et al. "3D Object Proposals Using Stereo Imagery for Accurate Object Class Detection". en. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.5 (May 2018), pp. 1259–1272. ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/TPAMI.2017.2706685. URL: https://ieeexplore.ieee.org/document/7932113/ (visited on 12/26/2023).

[25] Kyunghyun Cho et al. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. arXiv:1406.1078 [cs, stat] version: 3. Sept. 2014. URL: http://arxiv.org/abs/1406.1078 (visited on 12/27/2023).

[26] Shanta Chowdhury, Xishuang Dong, and Xiangfang Li. "Recurrent Neural Network Based Feature Selection for High Dimensional and Low Sample Size Micro-array Data". en. In: *2019 IEEE International Conference on Big Data (Big Data)*. Los Angeles, CA, USA: IEEE, Dec. 2019, pp. 4823–4828. ISBN: 978-1-72810-858-2. DOI: 10.1109/BigData47090.2019.9006432. URL: https:// ieeexplore. ieee. org/ document/ 9006432/ (visited on 01/14/2024).

[27] Junyoung Chung et al. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. en. Dec. 2014. URL: https://arxiv.org/abs/1412.3555v1 (visited on 01/12/2024).

[28] N. Cowan. "The magical number 4 in short-term memory: a reconsideration of mental storage capacity". eng. In: *The Behavioral and Brain Sciences* 24.1 (Feb. 2001), 87–114, discussion 114–185. ISSN: 0140-525X. DOI: 10.1017/s0140525x01003922.

[29] *Cross Entropy Loss: Intro, Applications, Code*. en. URL: https://www.v7labs.com/blog/cross-entropy-loss-guide,%20https://www.v7labs.com/blog/cross-entropy-loss-guide (visited on 12/18/2023).

[30] Shounak Datta, Vikrant A. Dev, and Mario R. Eden. "Developing QSPR for Predicting DNA Drug Binding Affinity of 9-Anilinoacridine Derivatives Using Correlation-Based Adaptive LASSO Algorithm". In: *Computer Aided Chemical Engineering*. Ed. by Antonio Espuña, Moisès Graells, and Luis Puigjaner. Vol. 40. 27 European Symposium on Computer Aided Process Engineering. Elsevier, Jan. 2017, pp. 2767–2772. DOI: 10 . 1016 / B978-0-444-63965-3.50463-3. URL: https://www.sciencedirect.com/science/article/pii/B9780444639653504633 (visited on 01/11/2024).

[31] Parth Rajesh Desai et al. *A Review Paper on Oculus Rift-A Virtual Reality Headset*. arXiv:1408.1173 [cs]. Aug. 2014. DOI: 10.48550/arXiv.1408.1173. URL: http://arxiv.org/abs/1408.1173 (visited on 12/24/2023).

[32] *Dropout and Batch Normalization*. en. URL: https://kaggle.com/code/ryanholbrook/dropout-and-batch-normalization (visited on 12/18/2023).

[33] *Dropout: Prevent overfitting*. URL: https://kharshit.github.io/blog/2018/05/04/dropout-prevent-overfitting (visited on 12/18/2023).

[34] *ECG Analysis-Based Cardiac Disease Prediction Using Signal Feature Selection with Extraction Based on AI Techniques | International Journal of Communication Networks and Information Security (IJCNIS)*. URL: https://www.ijcnis.org/index.php/ijcnis/article/view/5573 (visited on 12/30/2023).

[35] *Efficient Hyperparameter Optimization with Optuna Framework*. en. Apr. 2021. URL: https://broutonlab.com/blog/efficient-hyperparameter-optimization-with-optuna-framework/ (visited on 01/11/2024).

[36] *Electrocardiography - an overview | ScienceDirect Topics*. URL: https://www.sciencedirect.com/topics/earth-and-planetary-sciences/electrocardiography (visited on 01/10/2024).

[37] EliteDataScience. *Overfitting in Machine Learning: What It Is and How to Prevent It*. en-US. Sept. 2017. URL: https://elitedatascience.com/overfitting-in-machine-learning (visited on 12/18/2023).

[38] *Evaluation Metrics*. May 2019. URL: https://deepai.org/machine-learning-glossary-and-terms/evaluation-metrics (visited on 12/18/2023).

[39] *ExperienceTwin*. en-US. Jan. 2024. URL: https://www.experiencetwin.org (visited on 01/30/2024).

[40] *Eye tracker in VR - Learning Hub*. URL: https://varjo.com/learning-hub/eye-tracker-in-vr/ (visited on 12/07/2023).

[41] *Eye Tracking 101: What Is It & How Does It Work In Real Life? - Eyeware*. en-US. Running Time: 32 Section: 3D Eye Tracking. Mar. 2022. URL: https://eyeware.tech/blog/what-is-eye-tracking/ (visited on 12/09/2023).

[42] *Eye tracking: Your questions answered*. URL: https://varjo.com/vr-lab/eye-tracking-your-questions-answered/ (visited on 12/07/2023).

[43] Xiaoli Fan et al. "Assessment of mental workload based on multi-physiological signals". In: *Technology and Health Care* 28.Suppl 1 (), pp. 67–80. ISSN: 0928-7329. DOI: 10.3233/THC-209008. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7369076/ (visited on 01/30/2024).

[44] Hassan Ismail Fawaz et al. *Deep learning for time series classification: a review*. en. Sept. 2018. DOI: 10.1007/s10618-019-00619-1. URL: https://arxiv.org/abs/1809.04356v4 (visited on 01/12/2024).

[45] Stephanie Fishel, Eric Muth, and Adam Hoover. "Establishing Appropriate Physiological Baseline Procedures for Real-Time Physiological Measurement". In: *Journal of Cognitive Engineering and Decision Making* 1 (Dec. 2007). DOI: 10.1518/155534307X255636.

[46] *Frontiers | Pupil dilation as cognitive load measure in instructional videos on complex chemical representations*. URL: https:// www. frontiersin. org/articles/10.3389/feduc.2023.1062053/full (visited on 01/10/2024).

[47] Guanghui Fu et al. "Stable variable selection of class-imbalanced data with precision-recall criterion". In: *Chemometrics and Intelligent Laboratory Systems* 171 (Oct. 2017). DOI: 10.1016/j.chemolab.2017.10.015.

[48] Siheng Gao et al. "A Parallel Feature Fusion Network Combining GRU and CNN for Motor Imagery EEG Decoding". In: *Brain Sciences* 12.9 (Sept. 2022), p. 1233. ISSN: 2076-3425. DOI: 10.3390/brainsci12091233. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9496764/ (visited on 01/14/2024).

[49] *Gated Recurrent Unit Networks*. en-US. Section: Machine Learning. July 2019. URL: https://www.geeksforgeeks.org/gated-recurrent-unit-networks/ (visited on 12/27/2023).

[50] Consuela-Mădălina Gheorghe, Victor Lorin Purcărea, and Iuliana-Raluca Gheorghe. "Using eye-tracking technology in Neuromarketing." eng. In: *Romanian journal of ophthalmology* 67.1 (Mar. 2023). Place: Romania, pp. 2–6. ISSN: 2501-2533 2457-4325. DOI: 10.22336/rjo.2023.2.

[51] Austin Gibbs et al. "A universal, high-performance ECG signal processing engine to reduce clinical burden." In: *Annals of noninvasive electrocardiology : the official journal of the International Society for Holter and Noninvasive Electrocardiology, Inc* 27.5 (Sept. 2022), e12993. ISSN: 1542-474X 1082-720X. DOI: 10.1111/anec.12993.

[52] Margherita Grandini, Enrico Bagli, and Giorgio Visani. *Metrics for Multi-Class Classification: an Overview*. arXiv:2008.05756 [cs, stat]. Aug. 2020. DOI: 10.48550/arXiv.2008.05756. URL: http://arxiv.org/abs/2008.05756 (visited on 01/15/2024).

[53] Eija Haapalainen et al. "Psycho-physiological measures for assessing cognitive load". In: *Proceedings of the 12th ACM international conference on Ubiquitous computing*. Place: Copenhagen Denmark. ACM, Sept. 2010, pp. 301–310. ISBN: 978-1-60558-843-8. DOI: 10.1145/1864349.1864395. URL: https://dl.acm.org/doi/10.1145/1864349.1864395 (visited on 12/02/2023).

[54] Sandra G. Hart and Lowell E. Staveland. "Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research". In: *Advances in Psychology*. Ed. by Peter A. Hancock and Najmedin Meshkati. Vol. 52. Human Mental Workload. North-Holland, Jan. 1988, pp. 139–183. DOI: 10.1016/S0166-4115(08)62386-9. URL: https://www.sciencedirect.com/science/article/pii/S0166411508623869 (visited on 12/10/2023).

[55] Mahmoud Hassaballah et al. "ECG Heartbeat Classification Using Machine Learning and Metaheuristic Optimization for Smart Healthcare Systems". en. In: *Bioengineering* 10.4 (Apr. 2023). Number: 4 Publisher: Multidisciplinary Digital Publishing Institute, p. 429. ISSN: 2306-5354. DOI: 10.3390/bioengineering10040429. URL: https://www.mdpi.com/2306-5354/10/4/429 (visited on 12/09/2023).

[56] *Heart Rate Variability - How to Analyze ECG Data - iMotions*. en-US. Section: Consumer Insights. July 2019. URL: https://imotions.com/blog/learning/best-practice/heart-rate-variability/ (visited on 01/10/2024).

[57] Zejiang Hou. "Model and Data Efficiency in Deep Learning". en. In: (2022). Accepted: 2022-12-02T20:55:18Z Publisher: Princeton, NJ : Princeton University. URL: https://dataspace.princeton.edu/handle/88435/dsp01p8418r442 (visited on 01/11/2024).

[58] "http://spider.apa.org/ftdocs/rev/1994/april/rev1012343.html". en. In: ().

[59] *Individual Differences in Physiological Responses to Fearful, Racially Noxious, and Neutral Imagery - Marcia E. Sutherland, Jules P. Harrell, 1986*. URL: https://journals.sagepub.com/doi/10.2190/BJVV-1KK5-ATGW-RUY8 (visited on 12/10/2023).

[60] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. arXiv:1502.03167 [cs]. Mar. 2015. DOI: 10.48550/arXiv.1502.03167. URL: http://arxiv.org/abs/1502.03167 (visited on 12/18/2023).

[61] M. I. Jordan and T. M. Mitchell. "Machine learning: Trends, perspectives, and prospects". en. In: *Science* 349.6245 (July 2015), pp. 255–260. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aaa8415. URL: https://www.science.org/doi/10.1126/science.aaa8415 (visited on 03/02/2024).

[62] Kamen Kanev and Tomoyuki Sugiyama. "Design and simulation of interactive 3D computer games". In: *Computers & Graphics* 22.2 (Mar. 1998), pp. 281–300. ISSN: 0097-8493. DOI: 10.1016/S0097-8493(98)00038-7. URL: https://www.sciencedirect.com/science/article/pii/S0097849398000387 (visited on 12/24/2023).

[63] Sergios Karagiannakos. *Regularization techniques for training deep neural networks*. en. May 2021. URL: https://theaisummer.com/regularization/ (visited on 01/11/2024).

[64] R. E. Kleiger et al. "Time domain measurements of heart rate variability". eng. In: *Cardiology Clinics* 10.3 (Aug. 1992), pp. 487–498. ISSN: 0733-8651.

[65] Simeon Kostadinov. *Understanding GRU Networks*. en. Nov. 2019. URL: https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be (visited on 12/27/2023).

[66] Stephen D. Krashen. *Principles and practice in second language acquisition*. en. Reprinted. Language teaching methodology series. Oxford: Pergamon Press, 1984. ISBN: 978-0-08-028628-0.

[67] Krzysztof Krejtz et al. "Eye tracking cognitive load using pupil diameter and microsaccades with fixed gaze". en. In: *PLOS ONE* 13.9 (Sept. 2018). Publisher: Public Library of Science, e0203629. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0203629. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0203629 (visited on 01/10/2024).

[68] Sridhar Krishnan. *Biomedical Signal Analysis for Connected Healthcare*. Academic Press, June 2021. ISBN: 978-0-12-813173-2.

[69] Sangeeta Kumari and Nitish Polke. "Implementation Issues of Augmented Reality and Virtual Reality: A Survey". In: Jan. 2019, pp. 853–861. ISBN: 978-3-030-03145-9. DOI: 10.1007/978-3-030-03146-6_97.

[70] *L1 and L2 Regularization Methods, Explained | Built In*. en. URL: https://builtin.com/data-science/l2-regularization (visited on 01/11/2024).

[71] *Least Absolute Shrinkage and Selection Operator - an overview | ScienceDirect Topics*. URL: https://www.sciencedirect.com/topics/engineering/least-absolute-shrinkage-and-selection-operator (visited on 01/11/2024).

[72] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". en. In: *Nature* 521.7553 (May 2015). Number: 7553 Publisher: Nature Publishing Group, pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539. URL: https://www.nature.com/articles/nature14539 (visited on 12/09/2023).

[73] *Machine Learning Regularization Explained With Examples*. en. URL: https://www.techtarget.com/searchenterpriseai/feature/Machine-learning-regularization-explained-with-examples (visited on 01/13/2024).

[74] Darren George Mallery Paul. *IBM SPSS Statistics 26 Step by Step: A Simple Guide and Reference*. 16th ed. New York: Routledge, Dec. 2019. ISBN: 978-0-429-05676-5. DOI: 10.4324/9780429056765.

[75] Miss Priyanka Mayapur. "Detection and Processing of the R Peak". In: *IJIREEICE* 6.11 (Nov. 2018), pp. 36–44. ISSN: 23215526, 23212004. DOI: 10.17148/IJIREEICE.2018.6116. URL: https://ijireeice.com/wp-content/uploads/2018/12/IJIREEICE.2018.6116.pdf (visited on 12/03/2023).

[76] B. Mehlig. *Machine learning with neural networks*. en. Jan. 2019. DOI: 10.1017/9781108860604. URL: https://arxiv.org/abs/1901.05639v4 (visited on 01/12/2024).

[77] B. Mehlig. *Machine learning with neural networks*. arXiv:1901.05639 [cond-mat, stat]. Oct. 2021. DOI: 10.1017/9781108860604. URL: http://arxiv.org/abs/1901.05639 (visited on 03/17/2024).

[78] Gaurav Menghani. *Efficient Deep Learning: A Survey on Making Deep Learning Models Smaller, Faster, and Better*. arXiv:2106.08962 [cs] version: 2. June 2021. DOI: 10.48550/arXiv.2106.08962. URL: http://arxiv.org/abs/2106.08962 (visited on 01/11/2024).

[79] Hugo Mitre-Hernandez, Roberto Covarrubias Carrillo, and Carlos Lara-Alvarez. "Pupillary Responses for Cognitive Load Measurement to Classify Difficulty Levels in an Educational Video Game: Empirical Study". In: *JMIR Serious Games* 9.1 (Jan. 2021), e21620. ISSN: 2291-9279. DOI: 10.2196/21620. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7834946/ (visited on 01/10/2024).

[80] J. G. Moreno-Torres, J. A. Saez, and F. Herrera. "Study on the Impact of Partition-Induced Dataset Shift on $k$-Fold Cross-Validation". en. In: *IEEE Transactions on Neural Networks and Learning Systems* 23.8 (Aug. 2012), pp. 1304–1312. ISSN: 2162-237X, 2162-2388. DOI: 10.1109/TNNLS.2012.2199516. URL: http://ieeexplore.ieee.org/document/6226477/ (visited on 01/09/2024).

[81] Gregory R. Mostyn. "Cognitive Load Theory: What It Is, Why It's Important for Accounting Instruction and Research". In: *Issues in Accounting Education* 27.1 (Feb. 2012), pp. 227–245. ISSN: 0739-3172. DOI: 10.2308/iace-50099. URL: https://doi.org/10.2308/iace-50099 (visited on 01/15/2024).

[82] Alhassan Mumuni and Fuseini Mumuni. "Data augmentation: A comprehensive survey of modern approaches". In: *Array* 16 (Dec. 2022), p. 100258. ISSN: 2590-0056. DOI: 10.1016/j.array.2022.100258. URL: https://www.sciencedirect.com/science/article/pii/S2590005622000911 (visited on 01/11/2024).

[83] Fatma Murat et al. "Exploring deep features and ECG attributes to detect cardiac rhythm classes". In: *Knowledge-Based Systems* 232 (Nov. 2021), p. 107473. ISSN: 0950-7051. DOI: 10.1016/j.knosys.2021.107473. URL: https://www.sciencedirect.com/science/article/pii/S0950705121007358 (visited on 01/10/2024).

[84] Vrudhula K. Murthy et al. "Clinical Usefulness of ECG Frequency Spectrum Analysis". In: *Proceedings of the Annual Symposium on Computer Application in Medical Care* (Nov. 1978), pp. 610–612. ISSN: 0195-4210. URL:

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2231746/ (visited on 01/10/2024).

[85] Anuja Nagpal. *L1 and L2 Regularization Methods*. en. Oct. 2017. URL: https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c (visited on 01/11/2024).

[86] Andrei Nicolae. *PLU: The Piecewise Linear Unit Activation Function*. arXiv:1809.09534 [cs]. Sept. 2018. DOI: 10.48550/arXiv.1809.09534. URL: http://arxiv.org/abs/1809.09534 (visited on 03/17/2024).

[87] *OPTUNA: A Flexible, Efficient and Scalable Hyperparameter Optimization Framework | by Fernando López | Towards Data Science*. URL: https://towardsdatascience.com/optuna-a-flexible-efficient-and-scalable-hyperparameter-optimization-framework-d26bc7a23fff (visited on 01/11/2024).

[88] *Optuna: A hyperparameter optimization framework — Optuna 3.5.0 documentation*. URL: https://optuna.readthedocs.io/en/stable/ (visited on 01/11/2024).

[89] Fred Paas et al. "Cognitive Load Measurement as a Means to Advance Cognitive Load Theory". In: *Educational Psychologist - EDUC PSYCHOL* 38 (Mar. 2003), pp. 63–71. DOI: 10.1207/S15326985EP3801_8.

[90] Mateusz Paluchowski. *ECG_PLATFORM*. original-date: 2021-01-28T00:04:19Z. May 2023. URL: https://github.com/Gruschwick/ECG_PLATFORM (visited on 12/15/2023).

[91] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html (visited on 01/15/2024).

[92] Fabian Pedregosa et al. "Scikit-learn: Machine Learning in Python". en. In: *MACHINE LEARNING IN PYTHON* ().

[93] Lloyd Peterson and Margaret Jean Peterson. "Short-term retention of individual verbal items". In: *Journal of Experimental Psychology* 58.3 (1959). Place: US Publisher: American Psychological Association, pp. 193–198. ISSN: 0022-1015. DOI: 10.1037/h0049234.

[94] W. Piekarski, B. Gunther, and B. Thomas. "Integrating virtual and augmented realities in an outdoor application". en. In: *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*. San Francisco, CA, USA: IEEE Comput. Soc, 1999, pp. 45–54. ISBN: 978-0-7695-0359-2. DOI: 10.1109/IWAR.1999.803805. URL: http://ieeexplore.ieee.org/document/803805/ (visited on 12/24/2023).

[95] Marc Pomplun and Sindhura Sunkara. "Pupil dilation as an indicator of cognitive workload in human-computer interaction". In: *Proceedings of the International Conference on HCI* (Jan. 2003).

[96] *Practical Guide for Biomedical Signals Analysis Using Machine Learning Techniques - 1st Edition*. URL: https://shop.elsevier.com/books/practical-guide-for-biomedical-signals-analysis-using-machine-learning-techniques/subasi/978-0-12-817444-9 (visited on 12/03/2023).

[97] *Pupil dilation as an index of effort in cognitive control tasks: A review | Psychonomic Bulletin & Review*. URL: https://link.springer.com/article/10.3758/s13423-018-1432-y (visited on 01/10/2024).

[98] Hassan Ramchoun et al. "Multilayer Perceptron: Architecture Optimization and Training". en. In: *International Journal of Interactive Multimedia and Artificial Intelligence* 4.1 (2016), p. 26. ISSN: 1989-1660. DOI: 10.9781/ijimai.2016.415. URL: http://www.ijimai.org/journal/node/907 (visited on 12/17/2023).

[99] *Regularization in Deep Learning: L1, L2 & Dropout*. en. URL: https://www.e2enetworks.com/blog/regularization-in-deep-learning-l1-l2-dropout (visited on 01/11/2024).

[100] *Regularization in Machine Learning*. en-US. Section: Machine Learning. May 2019. URL: https://www.geeksforgeeks.org/regularization-in-machine-learning/ (visited on 01/11/2024).

[101] Antônio H. Ribeiro et al. "Automatic diagnosis of the 12-lead ECG using a deep neural network". en. In: *Nature Communications* 11.1 (Apr. 2020). Number: 1 Publisher: Nature Publishing Group, p. 1760. ISSN: 2041-1723. DOI: 10.1038/s41467-020-15432-4. URL: https://www.nature.com/articles/s41467-020-15432-4 (visited on 12/09/2023).

[102] Marc Rodemer, Jessica Karch, and Sascha Bernholt. "Pupil dilation as cognitive load measure in instructional videos on complex chemical representations". In: *Frontiers in Education* 8 (2023). ISSN: 2504-284X. URL: https://www.frontiersin.org/articles/10.3389/feduc.2023.1062053 (visited on 01/10/2024).

[103] Muammar Sadrawi et al. "Ensemble Genetic Fuzzy Neuro Model Applied for the Emergency Medical Service via Unbalanced Data Evaluation". en. In: *Symmetry* 10.3 (Mar. 2018). Number: 3 Publisher: Multidisciplinary Digital Publishing Institute, p. 71. ISSN: 2073-8994. DOI: 10.3390/sym10030071. URL: https://www.mdpi.com/2073-8994/10/3/71 (visited on 01/15/2024).

[104] Maria Sanchez-Vives and Mel Slater. "From presence to consciousness through virtual reality". In: *Nature reviews. Neuroscience* 6 (May 2005), pp. 332–9. DOI: 10.1038/nrn1651.

[105] Shibani Santurkar et al. "How Does Batch Normalization Help Optimization?" In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/hash/905056c1ac1dad141560467e0a99e1cf-Abstract.html (visited on 12/18/2023).

[106] Marko Sarstedt et al. "Progress in partial least squares structural equation modeling use in marketing research in the last decade". en. In: *Psychology & Marketing* 39.5 (2022). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mar.21640 pp. 1035–1064. ISSN: 1520-6793. DOI: 10.1002/mar.21640. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mar.21640 (visited on 03/04/2024).

[107] Richard M Shiffrin and Walter Schneider. "Controlled and Automatic Human Information Processing: II. Perceptual Learning, Automatic Attending, and a General Theory". en. In: ().

[108] Connor Shorten and Taghi M. Khoshgoftaar. "A survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* 6.1 (July 2019), p. 60. ISSN: 2196-1115. DOI: 10.1186/s40537-019-0197-0. URL: https://doi.org/10.1186/s40537-019-0197-0 (visited on 01/11/2024).

[109] Anupreet Kaur Singh and Sridhar Krishnan. "ECG signal feature extraction trends in methods and applications". In: *BioMedical Engineering OnLine* 22.1 (Mar. 2023), p. 22. ISSN: 1475-925X. DOI: 10.1186/s12938-023-01075-1. URL: https://biomedical-engineering-online.biomedcentral.com/articles/10.1186/s12938-023-01075-1 (visited on 12/03/2023).

[110] Sandra Śmigiel, Krzysztof Pałczyński, and Damian Ledziński. "ECG Signal Classification Using Deep Learning Techniques Based on the PTB-XL Dataset". In: *Entropy* 23.9 (Aug. 2021), p. 1121. ISSN: 1099-4300. DOI: 10.3390/e23091121. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8469424/ (visited on 12/09/2023).

[111] Alexander Soen and Ke Sun. *On the Variance of the Fisher Information for Deep Learning*. arXiv:2107.04205 [cs, stat] version: 3. Oct. 2021. DOI: 10.48550/arXiv.2107.04205. URL: http://arxiv.org/abs/2107.04205 (visited on 01/11/2024).

[112] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. ISSN: 1533-7928. URL: http://jmlr.org/papers/v15/srivastava14a.html (visited on 12/18/2023).

[113] Eli Stevens, Luca Antiga, and Thomas Viehmann. *Deep learning with PyTorch*. en. Shelter Island, NY: Manning Publications Co, 2020. ISBN: 978-1-61729-526-3.

[114] Julian Stieg, Peter Marks, and Lasse Gerrits. "UN-CODE: Software for Structuring and Visualizing Collective Decision-Making Based on Qualitative Data". en-US. In: 7.1 (July 2019). Number: 1 Publisher: Ubiquity Press, p. 25. ISSN: 2049-9647. DOI: 10.5334/jors.246. URL: https://openresearchsoftware.metajnl.com/articles/10.5334/jors.246 (visited on 12/12/2023).

[115] *Stratified KFold Tutorial | AnalyseUp.com*. URL: https://www.analyseup.com/python-machine-learning/stratified-kfold.html (visited on 01/09/2024).

[116] Lin Sun et al. "Feature selection using Fisher score and multilabel neighborhood rough sets for multilabel classification". In: *Information Sciences* 578 (Nov. 2021), pp. 887–912. ISSN: 0020-0255. DOI: 10.1016/j.ins.2021.

08.032. URL: https://www.sciencedirect.com/science/article/pii/S002002552100832X (visited on 01/11/2024).

[117] Lin Sun et al. "Feature selection using Fisher score and multilabel neighborhood rough sets for multilabel classification". In: *Information Sciences* 578 (Nov. 2021), pp. 887–912. ISSN: 0020-0255. DOI: 10.1016/j.ins.2021.08.032. URL: https://www.sciencedirect.com/science/article/pii/S002002552100832X (visited on 01/10/2024).

[118] Lin Sun et al. "Feature selection using Fisher score and multilabel neighborhood rough sets for multilabel classification". In: *Information Sciences* 578 (Nov. 2021), pp. 887–912. ISSN: 0020-0255. DOI: 10.1016/j.ins.2021.08.032. URL: https://www.sciencedirect.com/science/article/pii/S002002552100832X (visited on 01/11/2024).

[119] Lin Sun et al. "Feature selection using neighborhood entropy-based uncertainty measures for gene expression data classification". In: *Information Sciences* 502 (Oct. 2019), pp. 18–41. ISSN: 0020-0255. DOI: 10.1016/j.ins.2019 .05 .072. URL: https://www.sciencedirect.com/science/article/pii/S002002551930489X (visited on 01/11/2024).

[120] John Sweller. "Cognitive Load During Problem Solving: Effects on Learning". en. In: *Cognitive Science* 12.2 (1988). _eprint: https://onlinelibrary.wiley.com/doi/pdf/1 pp. 257–285. ISSN: 1551-6709. DOI: 10 . 1207 / s15516709cog1202 _ 4. URL: https://onlinelibrary.wiley.com/doi/abs/10.1207/s15516709cog1202_4 (visited on 01/15/2024).

[121] John Sweller. "Element Interactivity and Intrinsic, Extraneous, and Germane Cognitive Load". In: *Educational Psychology Review* 22 (June 2010), pp. 123–138. DOI: 10.1007/s10648-010-9128-5.

[122] John Sweller, Jeroen J. G. van Merrienboer, and Fred G. W. C. Paas. "Cognitive Architecture and Instructional Design". en. In: *Educational Psychology Review* 10.3 (Sept. 1998), pp. 251–296. ISSN: 1573-336X. DOI: 10.1023/A:1022193728205. URL: https://doi.org/10.1023/A:1022193728205 (visited on 01/15/2024).

[123] Szilvia Szeghalmy and Attila Fazekas. "A Comparative Study of the Use of Stratified Cross-Validation and Distribution-Balanced Stratified Cross-Validation in Imbalanced Learning". In: *Sensors (Basel, Switzerland)* 23.4 (Feb. 2023), p. 2333. ISSN: 1424-8220. DOI: 10.3390/s23042333. URL: https:

// www . ncbi . nlm . nih . gov / pmc / articles/ PMC9967638/ (visited on 01/09/2024).

[124] ODSC Team. *Optuna: An Automatic Hyperparameter Optimization Framework*. en-US. Sept. 2019. URL: https://odsc.com/blog/optuna-an-automatic-hyperparameter-optimization-framework/ (visited on 01/11/2024).

[125] ODSC Team. *Optuna: An Automatic Hyperparameter Optimization Framework*. en-US. Sept. 2019. URL: https://odsc.com/blog/optuna-an-automatic-hyperparameter-optimization-framework/ (visited on 01/11/2024).

[126] The pandas development team. *pandas-dev/pandas: Pandas*. Dec. 2023. DOI: 10.5281/zenodo.10426137. URL: https://zenodo.org/records/10426137 (visited on 01/15/2024).

[127] *The Essential Guide to Data Augmentation in Deep Learning*. en. URL: https://www.v7labs.com/blog/data-augmentation-guide,%20https://www.v7labs.com/blog/data-augmentation-guide (visited on 01/11/2024).

[128] *Top Data Augmentation Techniques: Ultimate Guide for 2024*. en-US. URL: https://research.aimultiple.com/data-augmentation-techniques/ (visited on 01/11/2024).

[129] Christoph Tremmel et al. "Estimating Cognitive Workload in an Interactive Virtual Reality Environment Using EEG". In: *Frontiers in Human Neuroscience* 13 (2019). ISSN: 1662-5161. URL: https://www.frontiersin.org/articles/10.3389/fnhum.2019.00401 (visited on 12/10/2023).

[130] *Ultimate Guidebook for Regularization Techniques in Deep Learning*. en. URL: https://www.turing.com/kb/ultimate-guidebook-for-regularization-techniques-in-deep-learning (visited on 01/11/2024).

[131] *Understanding LSTM Networks – colah's blog*. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (visited on 12/27/2023).

[132] Ashish Vaswani et al. *Attention Is All You Need*. en. June 2017. URL: https://arxiv.org/abs/1706.03762v7 (visited on 01/12/2024).

[133] Sara Ventura et al. "Immersive Versus Non-immersive Experience: Exploring the Feasibility of Memory Assessment Through 360° Technology". In: *Frontiers in Psychology* 10 (2019). ISSN: 1664-1078. URL: https://www.frontiersin.org/articles/10.3389/fpsyg.2019.02509 (visited on 12/24/2023).

[134] *VIVE Pro 2 | VIVE Business United States*. en-US. URL: https://business.vive.com/us/product/vive-pro2/ (visited on 01/30/2024).

[135] *VIVE Pro Eye*. en. URL: https://www.lib.ncsu.edu/devices/vive-pro-eye (visited on 01/30/2024).

[136] *What is Deep Learning? | IBM*. en-us. URL: https://www.ibm.com/topics/deep-learning (visited on 12/27/2023).

[137] *What is Deep Learning? \textbar IBM*. URL: https://www.ibm.com/topics/deep-learning (visited on 12/27/2023).

[138] *Why PyTorch Is the Deep Learning Framework of the Future*. en. Oct. 2019. URL: https://blog.paperspace.com/why- use- pytorch- deep- learning-framework/ (visited on 12/15/2023).

[139] D Wu and SC Guo. "An improved Fisher Score feature selection method and its application". In: *J. Liaoning Tech. Univ.(Nat. Sci.)* 38 (2019), pp. 472–479.

[140] Liangyong Yao, Kuangrong Hao, and Bing Wei. "Dynamic Feature Selection Model Based on Gated Recurrent Units for Time Series Forecasting". In: *2022 2nd International Conference on Computer, Control and Robotics (ICCCR)*. Mar. 2022, pp. 139–145. DOI: 10.1109/ICCCR54399.2022.9790206. URL: https:// ieeexplore . ieee. org / document / 9790206 (visited on 01/14/2024).

[141] Vishal Yathish. *Loss Functions and Their Use In Neural Networks*. en. Aug. 2022. URL: https://towardsdatascience.com/loss- functions- and-their-use-in-neural-networks-a470e703f1e9 (visited on 12/18/2023).

[142] Xue Ying. "An Overview of Overfitting and its Solutions". In: *Journal of Physics: Conference Series* 1168 (Feb. 2019), p. 022022. DOI: 10.1088/1742-6596/1168/2/022022.

[143] Gilsang Yoo, Hyeoncheol Kim, and Sungdae Hong. "Prediction of Cognitive Load from Electroencephalography Signals Using Long Short-Term Memory Network". In: *Bioengineering* 10.3 (Mar. 2023), p. 361. ISSN: 2306-5354. DOI: 10.3390/bioengineering10030361. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10044910/ (visited on 12/02/2023).

[144] Kyoung-Seok Yoo. “Motion prediction using brain waves based on arti- ficial intelligence deep learning recurrent neural network”. In: *Journal of Exercise Rehabilitation* 19.4 (Aug. 2023), pp. 219–227. ISSN: 2288-176X. DOI: 10.12965/jer.2346242.121. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10468292/ (visited on 01/14/2024).

[145] Aleksandra Zheleva et al. “CaliBrainVR: Using Psycho-physiological Mea- sures to Calibrate Virtual Reality Training”. en. In: *2023 15th International Conference on Quality of Multimedia Experience (QoMEX)*. Ghent, Belgium: IEEE, June 2023, pp. 123–126. ISBN: 9798350311730. DOI: 10.1109/QoMEX58391.2023.10178522. URL: https://ieeexplore.ieee.org/document/10178522/ (visited on 12/10/2023).

[146] Lim Jia Zheng, James Mountstephens, and Jason Teo. “Eye Fixation Ver- sus Pupil Diameter as Eye-Tracking Features for Virtual Reality Emo- tion Classification”. In: *2021 IEEE International Conference on Computing (ICOCO)*. Place: Kuala Lumpur, Malaysia. IEEE, Nov. 2021, pp. 315–319. ISBN: 978-1-66543-689-2. DOI: 10.1109/ICOCO53166.2021.9673503. URL: https://ieeexplore.ieee.org/document/9673503/ (visited on 12/07/2023).

[147] Yucong Zhou, Zezhou Zhu, and Zhao Zhong. *Learning specialized acti- vation functions with the Piecewise Linear Unit*. arXiv:2104.03693 [cs]. Apr. 2021. DOI: 10.48550/arXiv.2104.03693. URL: http://arxiv.org/abs/2104.03693 (visited on 03/18/2024).

[148] Wieske van Zoest, Stefan Van der Stigchel, and Mieke Donk. “Condi- tional control in visual selection”. en. In: *Attention, Perception, & Psychophysics* 79.6 (Aug. 2017), pp. 1555–1572. ISSN: 1943-393X. DOI: 10.3758/s13414-017-1352-3. URL: https://doi.org/10.3758/s13414-017-1352-3 (visited on 12/09/2023)