Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

THOMAS VIKSTRÖM

Developing measurement set-up for assessing cognitive workload with consumer-based EEG-device

DEGREE PROGRAMME IN WELFARE TECHNOLOGY

2022

Author(s)

Vikström, Thomas

Type of Publication

Master's thesis

Date

August 2022

Number of pages

70

Language of publication:

English

Title of publication

Developing measurement set-up for assessing cognitive workload with consumer-based EEG-device

Degree Programme

Welfare Technology

Abstract

A research study was conducted to find out if and how it might be possible to assess increased cognitive workload using a consumer-based EEG-device. Earlier research has established that it is possible to assess cognitive workload by using clinical or professional EEG-devices.

To reach the objectives of the study, a computer game was developed in Python using an iterative and incremental software development method. The game increases a user's cognitive workload or working memory by an adaptation of the well-known n-back psychological test. The developed game was successfully tested in a small experiment, using a consumer-based EEG-device, to be able to validate the functionality and usability of the game, and to collect EEG-data for further analysis. Data was processed, analyzed and visualized with spectrograms and a machine learning tool.

The study resulted in a versatile, configurable, and well-documented open-source n-back game which can be used as such or developed further by other Python-developers. Furthermore, the results from the experiment revealed that there are differences in the collected EEG-data between periods of rest and periods of higher cognitive workload. Identified limitations of the study as well as recommendations for further game improvements and research are included.

As a conclusion, based on the results of this study, it is possible to differentiate data from a consumer-based EEG-device between periods of rest and periods of higher cognitive workload.

Keywords

EEG, electroencephalography, EEG-device, n-back task, cognitive workload, working memory, software development, machine learning

FOREWORD

The author wants to acknowledge Professor Tarmo Lipping at the University of Tampere for providing the idea for this study as well as giving clear requirements for the software developed as part of it. Furthermore, Degree Programme Coordinator Anu Elo and Adj. Professor Sari Merilampi are acknowledged as they with their guidance have made this study so much better. Finally, heartful thanks go to the anonymous test persons who participated in the EEG-experiment!

CONTENTS

1 INTRODUCTION 6
2 PURPOSE AND AIM OF THE STUDY 7
2.1 Research questions 8
3 EEG, WORKING MEMORY, N-BACK TASK, MACHINE LEARNING, AND EARLIER RESEARCH 8
3.1 Electroencephalography (EEG) 9
3.1.1 EEG-devices 9
3.1.2 Utilizing EEG-signals in everyday situations 12
3.2 Working memory 13
3.3 N-back task 14
3.3.1 N-back levels 14
3.4 Machine learning 15
3.5 Earlier research 16
4 DEVELOPMENT METHODS AND TOOLS 18
4.1 Software development methods 18
4.2 Software development tools and programming languages 21
4.3 Machine learning tool 22
5 DESCRIPTION OF DEVELOPMENT PROCESS 24
5.1 N-back game development process 25
5.1.1 Requirement gathering 26
5.1.2 Initial functional requirements 27
5.1.3 Initial non-functional requirements 27
5.1.4 Purified requirements 28
5.1.5 Module design 30
5.1.6 Coding or programming 31
5.1.7 Functional testing 32
5.1.8 Game feedback survey 33
5.2 EEG-experiment 34
5.2.1 Equipment and tools used 34
5.2.2 Initiation 36
5.2.3 Game play settings 38
5.3 Machine learning process 39
6 RESULTS 41

6.1 N-back game 41
6.1.1 Iterations 41
6.1.2 The final product 44
6.1.3 Game feedback survey 45
6.2 EEG-experiment 46
6.2.1 Game play and observations 47
6.2.2 Spectrograms 49
6.2.3 Heart related data 52
6.2.4 Classification of n-game levels with machine learning 53
7 CONCLUSIONS AND DISCUSSION 55
7.1 How has the purpose been reached 55
7.1.1 Main success contributors 57
7.1.2 Conclusion 58
7.2 Limitations of this study 59
7.2.1 Quality of the process used in the study 59
7.2.2 Data quantity and quality limitations 60
7.2.3 Limited features in the n-back game 60
7.3 Recommendations 61
7.3.1 N-back game 61
7.3.2 Recommendations for further research 62
7.4 Contributions to research 63
REFERENCES

APPENDICES

1 INTRODUCTION

Measuring and utilizing signals from the human brain, both for research, but also for real life situations, has the recent years become a hot topic. Vance (2022) is describing how a BCI-implant (Brain-computer Interface) is used by patients to send WhatsApp messages and make online purchases just by using brain signals. The same device is now (July 2022) tested on a patient with ALS (amyotrophic lateral sclerosis), who has lost the ability to move and speak. The hope is that also this patient will be able to surf the web and communicate via email and text, again merely by thinking.

Non-invasive brain measuring devices are however even more common, and they too are being used for numerous purposes. Xu et al. (2022) outline how wired and wireless technology is used to examine attention of elementary school children while Lau-Zhu et al. (2019) propose that mobile brain measuring devices might open unprecedented possibilities for studying neurodevelopmental disorders. Yet another area under a lot of research is mental stress, and how it may contribute to health problems. Katmah et al. (2021) portray how brain signals might be used for assessing mental stress while Jun & Smitha (2016) designed an automatic stress recognition system based on brain signals.

Above mentioned use cases represent only an infinitesimal amount of all topics being researched or used in real life situations. While the solutions do differ with respect to the exact technology as well as overall implementation, they also have commonalities, one of them being the EEG (electroencephalography) technology used to measure brain signals. EEG-technology is based on devices that measure electrical signals near the surface of the brain, the signals are then further processed and analyzed with signal analysis tools.

This thesis will cover how a *consumer-based* EEG-device was used to assess cognitive workload by the help of a computer game coded by the researcher. It will also cover how the game was developed and how it can be configured and used for inducing

increased cognitive workload. The effort was at the same time contributing to an EU-financed brain health assessment and monitoring project led by a data-analytics and optimization research group at Tampere University. The project – "New technologies for the assessment and monitoring of brain health" – aimed to establish a research and development platform for new brain health and brain function monitoring systems and methods. It also planned to pilot an easy-to-use EEG-based brain health and brain function monitoring solution that would be seamlessly integrated to clothing. At the same time, with the large collaboration network of the team, competence was being developed to apply EEG measurements to versatile application areas, such as development of cognitive abilities, design of games, neurofeedback therapy, monitoring of work-related stress, and evaluating the effectiveness of rehabilitation and therapy. (Tampere University, 2022)

As main deliverable was developed a versatile n-back test game. This game was used in an experimental setting and the preliminary results of it are promising in that it seems very possible to assess cognitive workload with a consumer-based EEG-device.

2 PURPOSE AND AIM OF THE STUDY

The purpose of this study is to contribute to increased knowledge about how cognitive workload can be assessed through analyzing data coming from a consumer-based EEG-device. Consumer-based EEG-devices are more affordable and easier to use compared to professional and clinical devices. An easy-to-use device is wireless, and uses so called dry electrodes i.e., does not need a conductive gel or a saline solution to be applied to the electrodes. An affordable EEG-device was in this study defined to cost less than 400 €. Hence, these EEG-devices are, together with suitable software, expected to have a great potential in the future in measuring cognitive workload in scenarios where professional devices are not practical or economical to use.

As client for this study was the data-analytics and optimization research group at Tampere University.

The first aim of the study was to program a n-back memory game, and let the client use the program to assess cognitive workload from participants in an experiment. The n-back game is widely used to induce cognitive workload, to test working memory, and to analyze cognitive processes (Frost et al., 2021). Typically, the game will show a series of items, one at a time, and the test participant is expected to remember the last one, two, three, or sometimes even four items. This obviously puts a burden on the working memory of the participant.

The second aim was preliminary testing of the concept in order to assess the potential of the technology in this use case. The third aim was to, based on the above-mentioned test, define important aspects when collecting data from EEG-devices.

The client provided preliminary analyzed results from a few participants, while reporting statistically significant results was defined to be out of scope for this thesis.

2.1 Research questions

The research questions (RQ) set for this study were:

- RQ 1: How can cognitive workload be assessed with a consumer-grade EEG-device?
  - RQ 1a: What tools are needed?
  - RQ 1b: How can the tools be used?

3 EEG, WORKING MEMORY, N-BACK TASK, MACHINE LEARNING, AND EARLIER RESEARCH

Following subchapters describe the fundamental aspects of EEG, working memory, machine learning, and the n-back task, concluding with a discussion of earlier research.

3.1 Electroencephalography (EEG)

The human brain consists of billions of cells, roughly half of them are neurons, and the rest help and facilitate the activity of neurons. Neurons are densely interconnected via synapses, which act as activity gateways. Synaptic activity generates a tiny electrical impulse, and while the burst of a single neuron is difficult to reliably detect without direct contact with it, whenever thousands of neurons fire in sync, they however generate an electrical field which is strong enough to spread through tissue, bone, and skull. Eventually, this electrical field can be measured on the surface of the head (Farnsworth, 2021).

Electroencephalography, or EEG, is the physiological method of choice to record the electrical activity generated by the brain via electrodes placed on the scalp surface. As the voltage fluctuations measured at the electrodes are very small, the recorded data is digitized and sent to an amplifier. The amplified data can then be displayed as a sequence of voltage values, often using different graphs, like spectrograms. (Farnsworth, 2021). Soufineyestani et al. (2020) add that the amplified raw data is often filtered by bandpass filters or otherwise preprocessed before being used for further analysis.

3.1.1 EEG-devices

EEG-devices can be classified according to different categories, two of the main categories are:

- connection type: wired or wireless communication
- main target group: for professional/clinical use or for consumer use

The sensors in the devices measuring EEG-signals are also called electrodes, and the purpose of them are to provide a consistent electrical connection to the scalp of the person wearing the EEG-device. Soufineyestani et al. (2020) divide electrodes into three main types:

- Wet electrodes
  - Using a soft gel or saline solution to help lower the impedance between the electrodes and the skin. The electrodes need to be cleaned after each use which is typically done by using alcohol. A trained technician is often needed to set up EEG-devices using wet electrodes properly.
- Dry electrodes
  - No gel or saline solution necessary although connection time might be shortened by moistening the electrodes with water before use. Most consumer-based devices but also some professional devices use dry electrodes.
- Others
  - Electrodes that cannot clearly be categorized into wet or dry. For example, Enobio devices (Neuroelectrics, 2022a) have conductive solid gel materials in their electrodes.

Another way to categorize EEG-devices is according to main target customer group, consumers or professionals. A typical consumer device with four electrodes manufactured by Interaxon is shown in Figure 1 (left), and a professional device with 32 electrodes manufactured by Neuroelectrics (right).

Figure 1. Consumer EEG-headband - left (Interaxon, 2022a), professional EEG-device - right (Neuroelectrics, 2022b)

Differences between EEG-devices using wet or dry electrodes

Hinrichs et al. (2020) performed a comparison of the signal quality between dry wireless and wet wired EEG-devices, concluding that the quality is comparable. Some researchers however observed that, for activities demanding body movement like running/walking, wired wet sensors showed better performance. They further discuss that this seems to indicate that wet sensors may be more resistant to movement artifacts, although more research would be needed to fully understand which technology can provide more reliable data.

EEG-electrode positions

The American clinical neurophysiology society suggest guidelines for standard EEG-electrode position (Acharya et al., 2016). Figure 2 shows a commonly used version of the standard where the relative distance between an electrode and the underlying area on the skull is 10 %. The electrode location starts with a letter, where F = Frontal, C = Central, T = Temporal, P = Posterior and finally, O = Occipital. Depending on the position of the electrodes, they measure signals from different areas of the brain e.g., C1 is closely associated with the primary motor cortex, PO7 with the secondary visual cortex etc. (Scrivener & Reader, 2022).

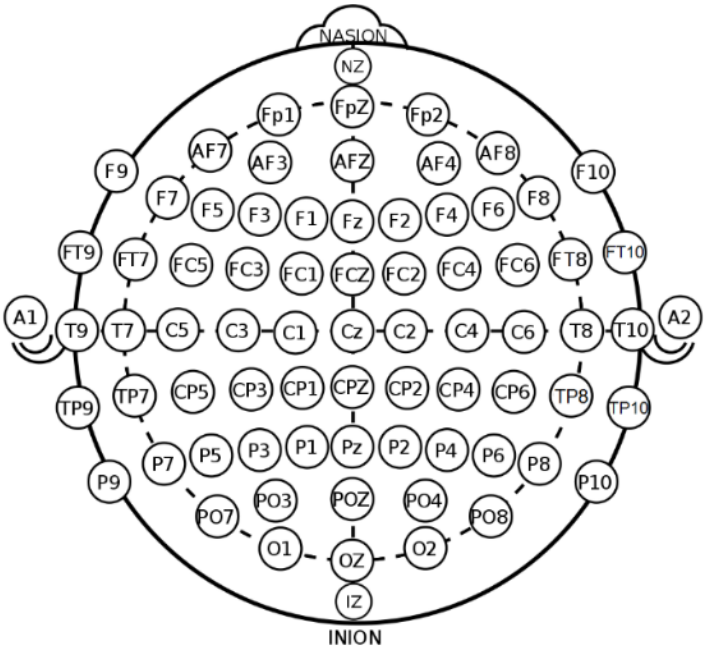


Figure 2. EEG-electrode positions (Wikimedia Commons, 2022)

Other types of sensors in EEG-devices

In addition to the EEG-electrodes, some EEG-devices include additional sensors e.g.:

- Photoplethysmography (PPG) sensor to determine blood flow, breathing rhythm, and heart rate
- Accelerometer/gyroscope to measure orientation, acceleration, and speed of head movements.

3.1.2 Utilizing EEG-signals in everyday situations

As mentioned in the introduction, EEG-signals are nowadays analyzed and utilized for numerous purposes, everything from helping paralyzed persons to communicate, to measuring attention or stress of school children or adults.

While *moderate* stress can be beneficial, *excessive* stress – especially during a longer time – can be detrimental to our health and in addition also decrease our performance. A significant reduction in performance can in worst case be catastrophic in time or quality critical jobs like e.g., air traffic controllers, pilots, nuclear plant operators etc. Stress can be measured with several methods: self-reports, behavioral measures, and medical/biological measures (Stanton et al., 2001). Of these methods, the two first ones are to some extent subjective in their nature and the last one more objective. Biological measurements are typically done by analyzing stress hormones or other markers in blood drawn from test subjects before, during, and after a stress inducing situation. Medical measurements in their turn can be done by different devices monitoring e.g.: blood-pressure, heart rate, heart rate variability, perspiration, brain signals, etc. Articles and conference papers covering stress monitoring using EEG-signals are abundant. For example, Hou et al. (2015) designed an EEG based stress monitoring system able to detect and classify stress into four, three, or two levels, and Al-shargie et al. (2016) were successfully discriminating stress from rest state in mental arithmetic tasks.

Related, but not identical, to stress is *mental* or *cognitive workload* which this study is focused on. Cognitive workload is defined as "the relative demand imposed by a particular task, in terms of mental resources required. Also called mental load; mental workload." (American Psychological Association, 2022). Niculescu et al. (2010) describe that when the load reaches a certain level of demand, people unconsciously appraise their abilities to meet the challenge, and only if the situation is considered as exceeding the available resources, then stress would appear. Their assumption is that even if related, these two factors can occur independently i.e., there is no compulsory relationship between them. Furthermore, stress might be produced by other variables that do not directly relate to cognitive load, such as noise, temperature, humidity, etc.

Measuring cognitive workload can to a large degree be done similarly as for stress: using self-reports, measuring behavioral, or using medical measures, including EEG. The *reasons* for assessing cognitive workload are also mostly the same as for measuring stress, one of them being reduced performance and the impact this may have.

3.2 Working memory

There are limits to how much a person can keep in mind at once. This so called working or short-term memory is typically limited to between three to five items. This capacity is important since mental tasks can be accomplished only with adequate ability to hold information as it is processed. Working memory is used in a variety of mental tasks, such as language comprehension, problem solving, and planning (for example determining the best order in which to visit the grocery, library, and pharmacy). Several studies indicate that working memory capacity varies among people, it may predict individual differences in intellectual ability, and it may change during the lifespan (Cowan, 2010).

Measuring a person's working memory capacity is quite straightforward as it only involves tallying the number of items the person is able to correctly remember. Whereas this can be performed completely in an analogue fashion, the focus is nowadays, especially in research, more often on digitalized solutions as they reduce

the administrative effort and the risk of human errors. One common method to assess working memory is to use the so-called n-back test which is introduced in next chapter.

3.3 N-back task

The n-back test or task is described by Frost et al. (2021) as a widely used method to analyze cognitive processes, showing a series of items, one at a time, and for every item asking the participant to indicate if the item is same as earlier. There are different difficulty levels where 0-back is the easiest, 1-back slightly more challenging etc. The n-back task was introduced by Wayne Kirchner (Kirchner, 1958). A *n-back game* is a digital version used, instead of previously used paper-based versions.

3.3.1 N-back levels

There are different versions of n-back tasks, the most elementary – which is used in this study – is to focus on remembering numbers only. Other versions include letters, symbols, sounds etc., or a combination of them, to remember.

Following is a description of the difficulty levels 0 – 2, levels 3 and up work similarly as level 2, but with one additional number to remember per level. The pictures show several numbers after each other, but in real situations only one number (or other item) is shown at a time, e.g. in Figure 3 the first number shown was 3, followed by 4, 2, 4 etc. The arrow above each picture, and which is pointing left, implies the time. One session can consist of one or several games, with a few or many numbers per game.

- Level 0, the user should react when a predefined target number is shown, in Figure 3 this number is 4.

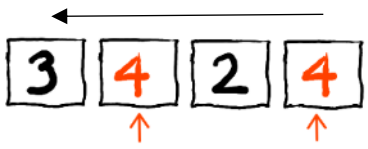


Figure 3. 0-back level

- Level 1, the user should react when previous number is repeated, in Figure 4 when the second number 3 and second number 4 is shown.

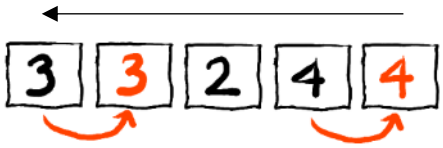


Figure 4. 1-back level

- Level 2, the user should react when the number shown is same as 2 numbers ago, in Figure 5 when number 3 and number 0 is shown the second time.

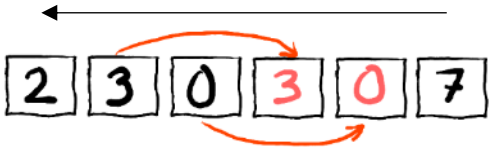


Figure 5. 2-back level

It is of course easy to "remember" items when several of them can be seen at once, like in the previous pictures. In the n-back task however, only one item is shown at a time with a short pause before it disappears and next item is shown. Thus, the participant needs to juggle several items in her/his working memory, and the higher the level, the more items need to be remembered. While in e.g. level 2 only two items need to be remembered at a time, they are constantly changing, and even this level can easily start to cause cognitive workload. Levels 3 and higher are significantly more challenging.

3.4 Machine learning

Machine learning was not in the original plan of this study. It was added after seeing the results of the EEG-experiment and concluding that with some further data processing and cleaning, more insights might be revealed related to the purpose of the study.

Machine learning as a paradigm includes many different sub-paradigms and algorithms, one of them neural networks. Hervé (1994) describes that neural networks are composed of basic units somewhat similar to brain neurons, and that these units are linked to each other by connections whose strength is modifiable as a result of a learning process or algorithm.

Figure 6 visualizes how a very simplified neural network works, where input in form of raw or transformed EEG-data is transferred to numerous *input layer* neurons. One neuron (input circles 1 – n) holds only one single value at a given time. Depending on the data, some of the connections to the neurons in the following *hidden layer* will through a mathematical algorithm become stronger and stronger for each iteration, in the picture this is indicated with thicker lines. These connections are also called *weights* as they have a numerical value, typically between -1.0 and +1.0. Finally, also some of the *weights* to the *output layer* will become stronger and predict a final output with some probability. In the picture, the connections to output B are stronger than to output 1, thus B would be the class or category the neural network has predicted from the EEG-measurement.

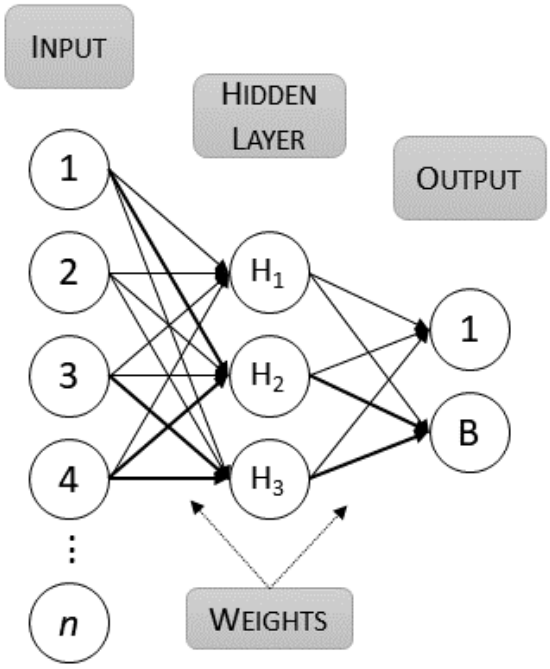


Figure 6. Visualization of a simplified neural network, own picture

3.5 Earlier research

The concept of analyzing EEG-data to measure cognitive workload is not novel. However, the concept of using *consumer-based* EEG-devices instead of professional or clinical EEG-devices, is according to an exploratory literature search not so much

researched. By for example using the search term “analyze EEG data cognitive workload” in Google Scholar, about 22100 references were found, approximately 5 % of them 30 years or older. However, when also including “consumer-based” in the search criteria, only 61 results were found. Without active journal subscriptions or separate article purchases, it was not possible to read the full text of most of the references, but both the subject and abstract for many of them suggest they do not cover same topic as this study.

Some of the studies found showed that it is possible to assess cognitive workload by measuring EEG-data. Kutafina et al. (2021) used a fully mobile setup to detect distinct levels of cognitive load, similarly Almogbel et al. (2019) found it possible to accurately detect cognitive workload of a vehicle driver, in this case using a consumer-based EEG-device. Ryu & Myung (2005) used a combination of physiological signals, one of them EEG, to determine mental effort needed for a task. Moren & Shipulin (2021) used a *single channel* consumer-based EEG-device to identify mental workload of test persons playing Tetris, and their analysis shows a significant enough correlation between subjective and mental workloads.

Of these studies, the one by Moren & Shipulin (2021) is similar to this study in the sense that both the Tetris and n-back games are used to induce an increased difficulty for each level, and hence supposedly also increase the cognitive workload. The difference on the other hand, is that while Moren’s study used a *single channel* device, this study used a *four channel* EEG-device which was thought to provide higher accuracy, albeit to a higher cost for the device (approximately 100 € for the single channel device vs. 270 € for the four-channel device).

Based on the exploratory literature search, the author argues that the idea of using consumer-based EEG-devices for assessing cognitive workload should be researched further.

4 DEVELOPMENT METHODS AND TOOLS

The following subchapters describe the development methods and tools, which were chosen for this thesis.

4.1 Software development methods

For software development there are a myriad of methods available, which can be categorized according to different criteria e.g., traditional vs modern, rigid vs flexible, etc. Despite the dissimilarities between the methods, they all consist of similar high-level steps, like research including gathering requirements, designing, planning, developing, testing, documenting, maintaining, etc. (Despa, 2014). Depending on software development method, the steps may though be named differently, they may be overlapping, or be in different order.

Despa (2014) describes that software development methodologies follow two major philosophies: heavyweight and lightweight. Heavyweight methodologies are suitable for projects having fixed requirements and the software complexity allows for detailed planning, tracking, and reporting. On the other hand, lightweight methodologies are suitable for projects having unclear requirements, or where they are likely to change. Furthermore, these methodologies are based on an incremental approach, they provide great flexibility and can easily adapt to change.

Before deciding on a suitable development method for this project, the main requirements for the method were defined considering the time available, persons involved, and project size. The requirements are described in Table 1.

Table 1. Main requirements for suitable development method

REQUIREMENT	DESCRIPTION
FLEXIBLE	The method should not be too formal or rigid
INCREMENTAL	Development done in shorter sprints, building up one module at a time, but also improving already implemented functionality.
CUSTOMER ORIENTED	Customer's feedback should be frequently collected
SCALABLE	The method should be scalable for this project

One of the more traditional methods the researcher has a lot of previous experience from, is the *waterfall* software development method seen in Figure 7 (Despa, 2014).

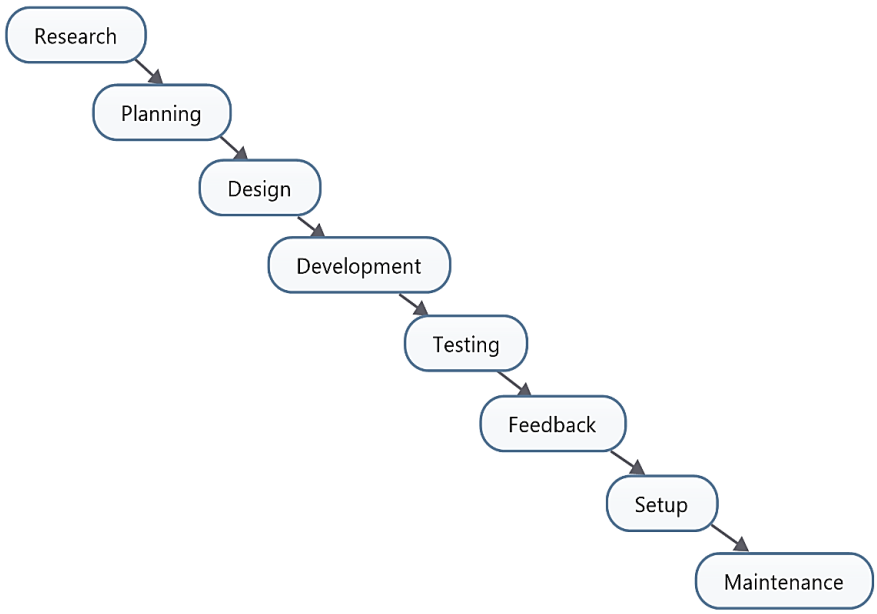


Figure 7. Waterfall development method, own picture, adapted from Despa (2014)

This method was however found to not work well for this project. One reason was that the waterfall method is quite rigid as it requires meticulous requirement gathering and planning before writing the first line of code, which typically leads to it taking a lot of calendar time. Another reason was that this method was not seen customer oriented as feedback is collected first after the software is completely developed and tested.

Instead, an *iterative and incremental* method was chosen that relies on developing one step at a time in the form of an expanding model, see Figure 8 (Despa, 2014).

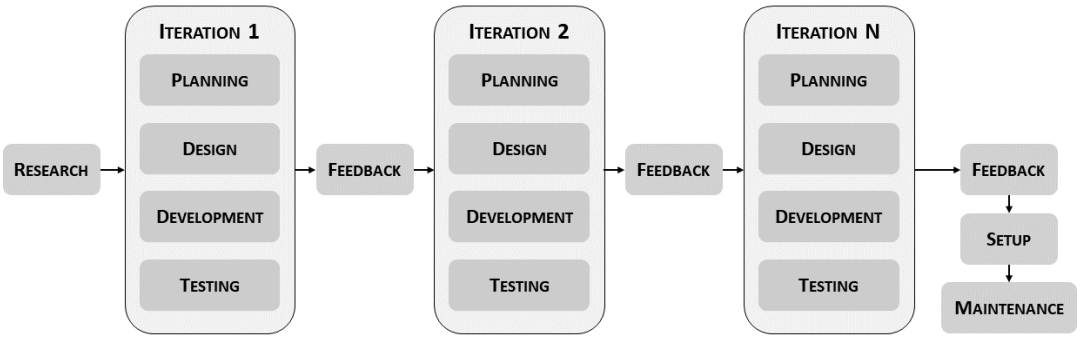


Figure 8. Iterative and incremental development method, own picture adapted from Despa (2014)

As seen in Figure 9, one step or iteration will include planning, designing, developing, and testing. The first iteration is in practice a prototype, and while it will have limited functionality, it will work as a proof of concept. It will also be the first time the client can see the game in action. After each iteration, feedback will be requested from the client to be able to fix possible issues, to adjust the overall design if needed, and to continue with next iteration. This is repeated until the game becomes fully functional and meets all the requirements.

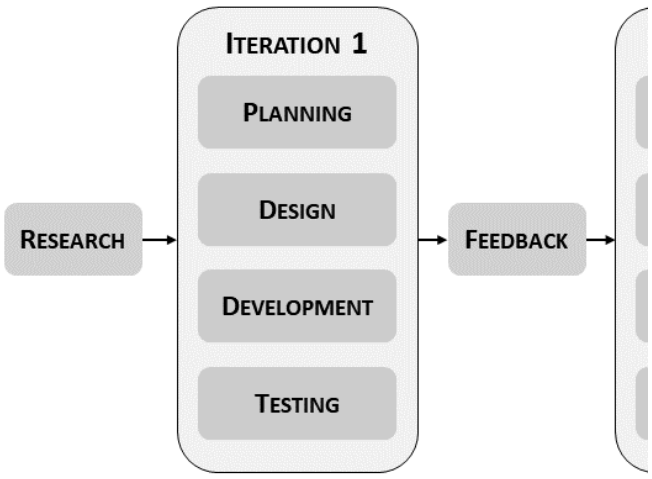


Figure 9. Iterations, own picture, adapted from Despa (2014)

The iterative and incremental method is quite similar to the nowadays popular Scrum development method, where iterations are named sprints. The major difference is that in the Scrum method there is always a team, having daily 15 minutes so called stand-up meetings, whereas this is not imperative in the iterative and incremental method (Despa, 2014).

One risk with the chosen development method is that the scope might increase once the client tests each iteration and might come up with completely new or majorly changed requirements. While adjusting for *minor* requirement changes is actually an inherent part of the iterative and incremental method, the risk of *major* requirement changes was in this project mitigated or at least minimized, by clearly stating what was in scope and what was out of scope for the game.

4.2 Software development tools and programming languages

For developing a n-back game two main routes were identified, a) use an existing platform for performing different psychological tests, or b) code the game more or less from scratch. The existing test platforms found might in the long run have advantages for longitudinal research done by same team, but drawbacks are that many of them are expensive and might take lot of work to set up the first time. Obviously coding a game from scratch would also take some amount of time, hence existing open-source program code was searched for and investigated. Many programs written in the Python programming language were found, and while two of them were very good candidates for this project, they were rejected due to no or very poor documentation. Without proper documentation, it was seen to be too time consuming to understand the code and further develop the programs. In the end, it was decided to code the game from scratch but, if possible, to use open-source modules for the menu system.

There are scores of programming languages, and most of them could have been used for this project. Two indexes over programming language popularity, Tiobe (2022) and PYPL (2022), however list Python as most popular (see Figure 10), and this was partly why Python was chosen as it was expected to be much easier to get help with a widely used programming language compared to a more uncommon one. Another even more important reason for why Python was chosen, was the researcher's experience with it.

Worldwide, Jun 2022 compared to a year ago:

Rank	Change	Language	Share
1		Python	27.61 %
2		Java	17.64 %
3		JavaScript	9.21 %
4		C#	7.79 %
5		C/C++	7.01 %

May 2022	May 2021	Change	Programming Language	Ratings	Change
1	2	▲	Python	12.74%	+0.86%
2	1	▼	C	11.59%	-1.80%
3	3		Java	10.99%	-0.74%
4	4		C++	8.83%	+1.01%
5	5		C#	6.39%	+1.98%

Figure 10. PYPL (2022) index - left, Tiobe (2022) index - right

As development tool for Python can be used everything from the very simplest text editor to a full-fledged IDE (Integrated Development Environment). A good development tool supports writing and understanding code with syntax coloring and helps reducing errors by signaling possible issues. The selected IDE was the free Microsoft Visual Studio Code, which is very popular and user-friendly, supporting basically all programming languages, including Python.

4.3 Machine learning tool

The original research plan did not include using machine learning on the collected EEG-data. It was however decided to investigate if it is possible to use machine learning for classifying the data to be able to distinguish between the different game levels. So, together with the analyzed spectrograms, machine learning was used to triangulate the findings.

As it was known beforehand that the data collected would not be nearly enough for this purpose, and as was found out later, also of varying quality, it was likewise here clear that possible results would not be statistically significant.

There are many different variants of neural networks, and partially or completely new ones are frequently developed. While it is possible to build neural networks entirely from scratch, there are nowadays many tools available that provide simple to use graphical interfaces, at the same time hiding the neural network engine and other technicalities that are of little or no interest to a majority of the users.

The machine learning tool that was used for this project was Edge Impulse, a web-based platform that can be used for free, although with some resource and data size limitations. For larger projects, an enterprise version is available. The reason for why this tool was chosen, was that the researcher has extensive experience of using it for machine learning purposes in real situations and for educational purposes.

5 DESCRIPTION OF DEVELOPMENT PROCESS

Figure 11 shows an overview of the main steps taken in this study, the steps are commented below.

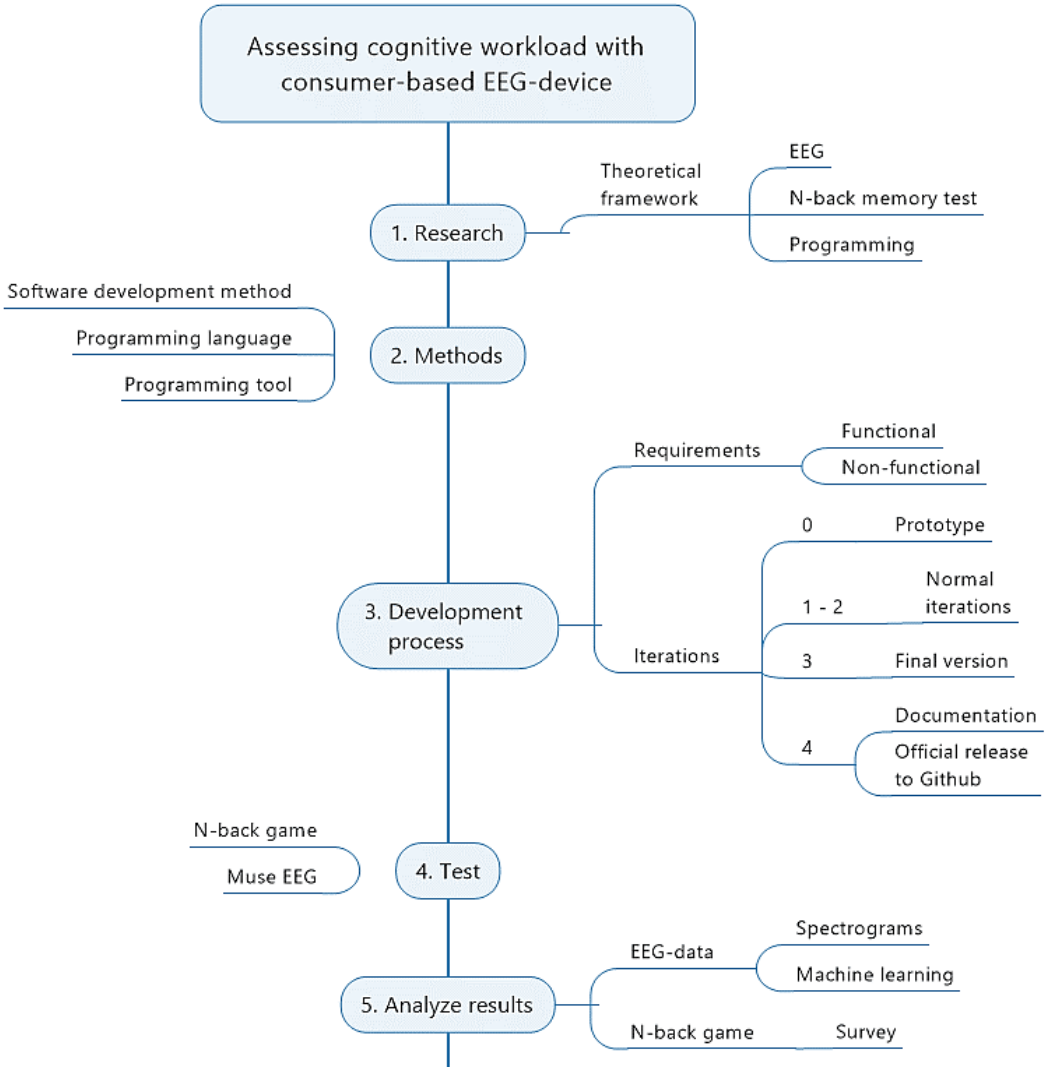


Figure 11. Main steps of the study.

1. Earlier research show that it is indeed possible to assess cognitive workload with *professional* devices measuring EEG-signals from the brain. What has not been exhaustively researched is using a *consumer-based* EEG-device for the same purpose. See chapter 3.

2. After evaluating existing solutions for the n-back test, it was decided to program it from scratch in the Python programming language, using an iterative and incremental method. See chapter 4.
3. The n-back memory game was by the researcher developed using best practices and according to functional and non-functional requirements. See chapter 5.1.
4. The game was used in a small experiment with only three participants wearing a consumer-based EEG-device. See chapter 5.2
5. The results from the experiment were analyzed using spectrograms and later with machine learning. See chapter 6.

5.1 N-back game development process

As stated in chapter 4.1, an iterative and incremental software development method or process was chosen. This process consists of iterations that all are small or even tiny development projects in themselves, with planning, designing, developing, and testing each deliverable from the iteration. The iterations build upon previous ones, extending and improving the software with each. A visual description of the theoretical process

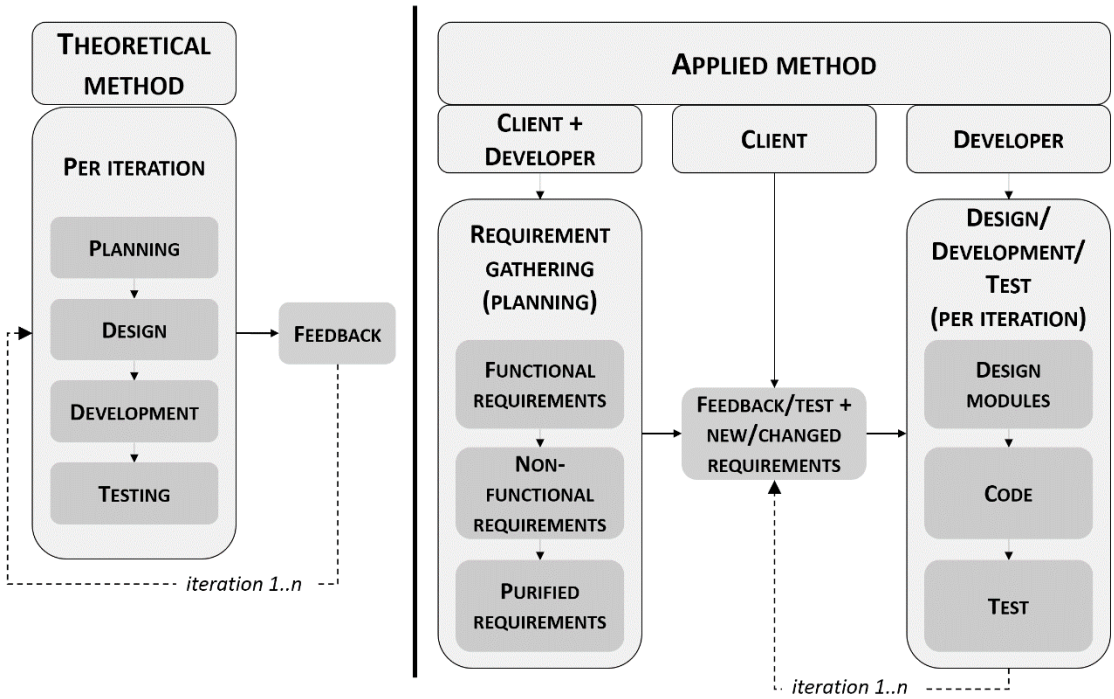


Figure 12. Theoretical method (left), applied method (right). Own picture.

for each iteration is shown at the left in Figure 12. The applied process is shown at the right, including who was the main responsible in the different steps. As can be seen, the main steps (planning, design, development, testing, and feedback) all map between the theoretical and the applied process.

Following chapters describe in more detail *how* this process was utilized in developing the game. The first chapters (5.1.1 – 5.1.4) explain the planning and design steps which mainly consisted of detailing the requirements, while chapters 5.1.5 - 5.1.7 describe module design, coding, and testing.

5.1.1 Requirement gathering

In this project the planning and designing steps mainly consisted of exploring and purifying the requirements as well as describing to the client *how* the game is planned to work.

General and very high-level requirements of most computer programs are that they should meet the required goals, be user-friendly, have high enough quality, and continue to work for some – sometimes vaguely – specified time. The magnitude and priority of these high-level requirements are obviously different between different types of programs e.g., a nuclear plant controller system has very divergent requirement prioritizations compared to an entertainment game.

Requirements for software are further characteristically defined into functional and non-functional requirements (Dabbagh & Lee, 2014). The first ones are defining how the program is supposed to *work*, and the latter ones are defining how the program is or can be *supported*, that is, how it can *continue* to work. Inexperienced software procurers might prioritize functional requirements too high over non-functional ones and fall for a seemingly lucrative price tag. They might not comprehend that a software not being supported, or being possible to upgrade, will soon de facto cease to work due to outside factors, unless a support or upgrade package is purchased.

Most of the functional requirements were given by the client or in discussion with them, while the non-functional requirements were mainly defined by the researcher based on decades of experience with developing and supporting software.

5.1.2 Initial functional requirements

Initial functional requirements for the game were quite few as well as broad:

- Levels 0 – 2 should be supported
  - level 0 = action expected when a target number is shown
  - level 1 = action expected when shown number is identical to previous
  - level 2 = action expected when shown number is identical to 2 numbers ago
- Relaxation period before/between games
- Option to request test participant to indicate if number shown is identical – as per level set – or not identical. In practice the participant is expected to click e.g., left mouse button if the number is identical, or right button if it is not identical.
- Option to show pictures to raise emotional stimulus while playing the game. These pictures would be from a picture library using stimulus categories such as fear, anger, love, happiness etc.

5.1.3 Initial non-functional requirements

Based on the intended user group, three high level non-functional requirements prioritized for this game were usability, flexibility, and maintainability. With usability in this context, was implied that the user should be able to use the game without having to read any separate instructions, and that the game flow should be logical and without surprises. Flexibility was expected to be needed for the researching project to adjust the game for different scenarios, without having to change the code itself. While maintainability – in this context how easy features can be changed or added – was not an explicit requirement from the client, it was seen to be of high importance so the

software could live for longer than for this project only, thus contributing to research and to the concept of open science.

5.1.4 Purified requirements

As it was seen imperative to clearly define what would be in scope and out of scope for the project, the given functional and non-functional requirements were further purified and added to. The resulting agreed upon requirements are found in Table 2 and Table 3. However, worth noting is that the chosen software development process is allowing new or changed specifications in each iteration, after feedback is received from the client.

Table 2. Functional requirements

REQUIREMENT	SCOPE
User ID to be used for identifying test participants, alphanumerical	In scope
Choice between numbers or letters (or combination of them?)	Numbers in scope, letters out of scope
Choice of level 0-back to n-back (in theory n can be infinite, in practice 4 might be a maximal level from human memory point of view)	Levels 0 – 4
Choice between different background/foreground colors	Out of scope
How many “games” to play, one game is a set of slides (= numbers/letters)	In scope
How many slides to play in each game	In scope
How long to show a slide, how long pause between	In scope
How long the relaxation time between games should be, a sound will signal when this time has ended and after which next game will shortly commence.	In scope
Option to show background images to possible raise different emotions	In scope only if complexity is low to medium
Current time shown in menu, hh:mm:ss (this to secure the computer and possible external device used to capture EEG-data have same time)	In scope
Recording numbers shown and timestamps (millisecond accuracy) for user actions to a log file, preferably in CSV-format to be able to easily use Excel for analysis.	In scope

Showing short feedback to the user, can be absolute scores and/or percentage of numbers remembered correctly	In scope
Possibility to use mouse or keyboard when reacting to a number	In scope

Table 3. Non-functional requirements

CATEGORY	REQUIREMENT	SCOPE
USABILITY	Menu system, possible to use mouse and/or keyboard when navigating	In scope
	Settings menu where *all* agreed settings can be done. Code changes/editing text files should not be needed to change settings. The number of settings is expected to grow during the iterations.	In scope
	Documentation for users The software should be so easy to use and understand that no detailed user instructions should be needed. Settings and different game modes will be described.	In scope
FLEXIBILITY	It should be easy for a *developer* to add new settings to be used in the game. Only minor code changes should be needed.	In scope
MAINTAINABILITY	It should be easy to understand the code and to add features to it. This will be done through modularizing the code and documenting it very thoroughly.	In scope
SUPPORTABILITY	The author is not necessarily continuing to develop or support the software after the project has ended. To allow anyone to support it, it will be open-sourced and stored in a public repository.	In scope
	Documentation for developers Instructions will be provided for how new settings can be added. All other developer documentation is provided through well-documented modularized code structure	In scope
	Installation program	Out of scope
	Supporting any other platform than a normal Windows 10 computer. Might still work on Macs, Linux, Raspberry PI computers etc., but will not be tested.	Out of scope

These purified requirements were provided to the client for approval after which they formed basis for the final game design.

5.1.5 Module design

Among best practices in programming are to modularize the code into functions or procedures where each function has clearly defined inputs and outputs. This makes the program significantly easier to implement, test, document, and maintain in the future (Lutz, 2013). These functions or procedures can be called *isolated black boxes* (see Figure 13), where each box (= function) typically takes some input, does something with it, and then produces output.

Figure 13. Isolated black box with inputs and outputs

It is called an isolated black box as everything – and everyone – outside the box should not need to know *how* it works, just *what* it does. In many programming languages, also in Python, modules and functions are used to represent these black boxes. Taking modularization to the extreme, *all code* – at least in Python – can be represented by modules and functions, even the main structure. Adding functionality is always done through modules and functions, and the only things needing testing are that the functions receive proper input, do what they should, and produce correct output. Another way to look at this, is to see the black box as a piece in a jigsaw puzzle, tests are only done on the piece itself, and on its interfaces to the outside world, see Figure 14.

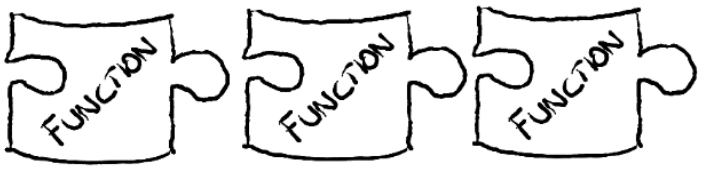


Figure 14. Black boxes or functions, with interfaces to the outside world

As an example of how modularization was implemented in this project, Figure 15 shows how the function **`play_level`** is called, as input taking the difficulty level to be played through the variable **`game`**, and the output produced from it will be put in the variable **`game_score`**. This is the calling interface, analogous to the protruding parts of jigsaw puzzle pieces.

```
for game in games:                          # play every game found in the list
    game_score = play_level(int(game))      # playing the game
```

Figure 15. Function calling interface

The function **`play_level`** itself is as input receiving the **`level`** to be played as portrayed in Figure 16. As this particular function, playing a game level, is quite long and itself calling several subfunctions, it is not further described here and remains as a black box.

```
# Playing a level ~~~~~
def play_level(level):
```

Figure 16. Function header

And finally, in Figure 17 the function is producing and returning output, in this case in form of scoring information to be put in the variable **`game_score`** earlier shown in Figure 15.

```
return [game_score, max_score, breaker]
```

Figure 17. Function output

5.1.6 Coding or programming

The coding or programming step in this study consisted of writing the code itself inside the functions, calling other functions, and documenting what each code row or snippet does. A code snippet is shown in Appendix B, including a link to the public code repository.

5.1.7 Functional testing

Software testing is a separate discipline within computer science and the main purpose of it is to secure that a system is working as it should, and that possible user or technical errors are handled properly. Apart from testing the software functionality itself, other areas are also often tested e.g., performance, availability, usability, etc. Testing should never be underestimated and will in some situations require much more time than writing the code itself, especially for systems where it is of utmost importance to have as close to 100 % quality as possible e.g., in aircraft or nuclear plant controlling software. Ensuring this high quality is not easy, even with a lot of testing, but by *planning quality* into the system during development, it will at least be easier. Finally, it is often stated that a completely 100 % bug-free software of size cannot be written, as the only way to ensure it is bug-free is to have automated testing. As the automated testing platform is another software, it can obviously also have bugs, ad infinitum! Thus, sometimes it is enough to secure no critical bugs are found, or to have redundant systems or fail-safe modes to mitigate or at least minimize the effects of potential bugs.

As this project was fairly compact time wise, and having quite clear requirements, it was seen more important to focus on the testing itself than to spend time on finding the optimal test methodology.

One way to reduce the effort needed to test, is to *prevent bugs* (= plan quality) by modularizing the code into Python modules and functions, as earlier described. The subsequent needed testing followed in brief these steps:

- Every new or added function was defined to have clear inputs and/or outputs, some functions also performed an action like e.g., showing a number. Testing was done to verify the input received from previous step was correct, the function took correct actions, and it produced correct output to next step. Almost all bugs and shortcomings were found in this testing.
    - This is an adaptation of the Black-box testing described by Jiantao (1999) as being a method emphasized on executing the functions and verifying their input and output data.

- When major code changes were done, or when many new features had been added, the whole program was tested, in practice this was done at least once per iteration. This was to further ensure that nothing had technically broken, but also that the game fulfilled the client requirements. It is not enough that a system is without technical bugs, it should also do what it is expected to!
- Each iteration was sent to the client for feedback and testing. While no technical bugs were found by the client, they did give suggestions to improve the game, some of which had verbally been touched upon earlier, but not been written down as requirements. As all these suggestions were of minor size, it was possible to include them in the scope without risking the schedule or quality.

5.1.8 Game feedback survey

To assess if the developed game was fulfilling the set requirements, and how well it could be used for assessing cognitive workload, a short questionnaire was sent to the research group. The questions were either quantitative or qualitative in their nature, one of the quantitative questions was an adaptation of the NPS (Net Promoter Score) introduced by (Reichheld, 2003). Simplified, the idea with NPS is to have one single key performance indicator (KPI) describing customer satisfaction and loyalty with a product or service, where a higher score indicates having more promoters than detractors. As this questionnaire was sent to only one person, representing the client, a statistically significant NPS could not be calculated and it would remain indicative only.

The questionnaire, which is found from Appendix 3, was sent through e-mail to the person leading the experiment.

5.2 EEG-experiment

Purpose of the small EEG-experiment was to gather feedback related to the n-back game, as well as to get some initial results from measuring EEG-signals from a consumer-based device.

The experiment sessions were executed May 6th, 2022, at UC Pori with three adult volunteers. The EU research project took care of the recruitment of the healthy volunteers who were part of the development team. They were aware of the purpose of the experiment and knew how the data would be used, and that they would stay anonymous. The experiment that was done, was not invasive and did not cause harm to the participants.

Before the experiment session started, the three levels were explained on a whiteboard to the participants who also were able to try the levels in the n-back game itself.

The author observed and documented the experiment in a detailed way to be able to link observations to collected EEG-data, and furthermore to reflect on how well the developed n-back game fulfilled the criteria and requirements being set for it e.g., how easy is it to use, how flexibly can it be configured etc.

Following chapters describes how the experiment was performed, what tools were used, observations done by the author, and comments received from participants in the experiment.

5.2.1 Equipment and tools used

EEG-device

As EEG-device was used Muse S (see Figure 18) which is a consumer headband having the main purpose to support meditation and sleep through measuring EEG-signals. Apart from the electrodes and essential electronics, everything else is made by textile which makes the headband more comfortable to wear than the Muse-headbands

made of plastic. One potential drawback with the Muse S device compared to its siblings, is that it is a bit finicky to get good connections with, especially from the electrodes behind the ears. However, it is expected to become easier with time after having gotten used to setting it up properly. On the other hand, due to the softness and elasticity of the Muse S device, it is likely to be more comfortable than many other ones when having to wear it for a longer period.

Figure 18. Muse S headband (Interaxon, 2022b)

The approximate electrode locations this device use is the frontal AF7 and AF8 which rest at the forehead, temporal TP9 and TP10 which rest behind the ears, and finally the forehead FpZ as reference sensor. See Figure 2 in chapter 3 for an overview of the locations.

Computer

An ordinary laptop with the Windows 10 operating system was used. The laptop was connected to an external 24-inch monitor such that the screen was duplicated i.e., both monitors displayed identical content. This setup allowed the researcher to observe from a larger distance compared to if only one monitor had been used. A normal keyboard and mouse were connected to the computer.

Connection method

Current Muse headbands on the market use the Bluetooth 4.2 standard to connect to smartphones or computers. While the Muse devices have a MicroUSB port, it can only be used for charging, not for transferring data.

Software

The following software was used:

- N-back game
- Petal Metrics for streaming data from the EEG-device to the computer being used for playing the game. The reason for using same computer for both was that it was seen very important to have identical data timestamps between the game and EEG-data. Petal Metrics saves the data into separate CSV-files for EEG-data, heart related data, movement data, and telemetry data.
- HeartPy, an open-source Python program for analyzing heart related data from PPG-sensors
- Mind Monitor app (Android/iOS) for verifying good signal before game start.

5.2.2 Initiation

The computer and monitor were placed on an electrically adjustable desk, this allowed comfortable positions for the test persons. The desk was from before set up for working in a standing position, and none of the experiment participants asked for it to be lowered to be able to sit on a chair.

To reduce the time to connect to the Muse device and to acquire good signals, the electrodes of the device were watered before each test. Still, it took three to four minutes before the signals had stabilized enough. The Mind Monitor app was used to monitor the connection and signals before connecting the device to the computer, this is shown in Figure 19.

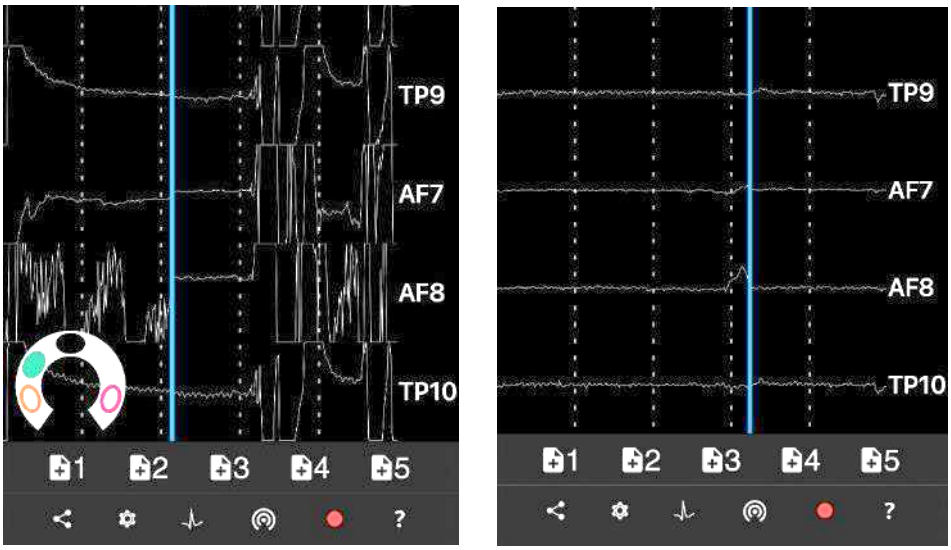


Figure 19. Poor connection (left), excellent connection (right)

Signals from the four electrodes are in the picture indicated by the horizontal light gray lines, and the measurement point in time by the blue vertical line. The left picture shows a horseshoe where the upper right blank area is indicating no or poor connection from the frontal right electrode, this is also visible in the jittering signals. The right picture again shows excellent connection with basically no jitter.

The reasons for why it took a long time to get good connection was not further investigated, but electrical interference from the relatively close fluorescent lamps or other appliances might have affected. When using the Mind Monitor app, it is in fact possible to choose the power line frequency used in the country, in Finland and other countries with 220 – 240 V this frequency is 50 Hz while in other countries it is 60 Hz. However, in Petal Metrics which was used for streaming data, this adjustment is not available, hence reducing electrical interference and other noise needs to be done afterwards.

To find out how this electrical interference might manifest itself, an experiment was later conducted where the author first was indoors, and later outdoors, at both times wearing a Muse device and using the Mind Monitor app. The spectrograms in Figure 20 show the results of this experiment where the left picture is when being indoors, and the right picture when being outdoors. The interference around 50 Hz as indicated by the arrow, and visible as a vertical red line, has all but disappeared in the right picture. The other red areas are normal brain signals. Thus, electrical interference is a factor to be aware of as most EEG-experiments are performed indoors and often close to power outlets, lamps, etc.

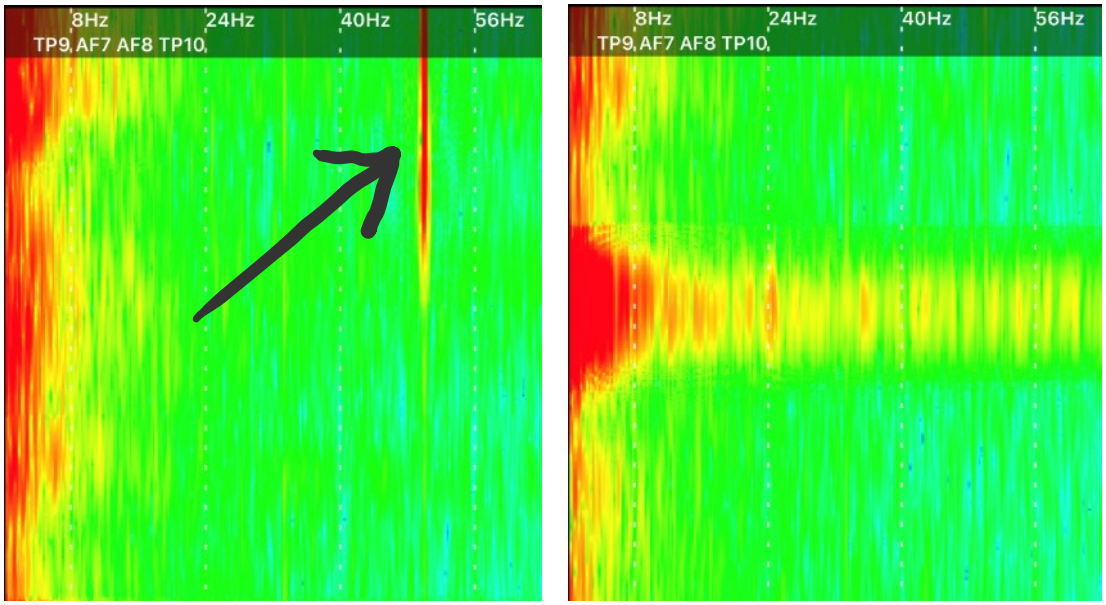


Figure 20. Power line interference: indoors (left) and outdoors (right)

5.2.3 Game play settings

Table 4 lists the main settings that were made in the n-back game before the experiment sessions. Essentially the game was set up, so each participant played seven games with 50 numbers shown in each game, leading to a total session time of approximately 15 minutes, including the 30 second relaxation time between each game. Apart from the listed settings, additional options related to e.g., screen resolution, font sizes etc. are available and can be found in Appendix 1.

Table 4. Essential n-back game test settings

NAME	SETTING	DESCRIPTION
RANDOM MULTI GAMES, HOW MANY TO PLAY?	7	7 games will be played after each other
RANDOM MULTI GAMES, LEVELS?	0 – 2	Levels between 0 and 2 will be randomly chosen
0-BACK TARGET NUMBER	4	If level 0 is played, number 4 will be the target number that the test person should remember and react on.
BREATHING TIME	30	Test persons are expected to relax with closed eyes for 30 seconds before each game. After 30 seconds a chime will sound to indicate the level can be started.
SHOWING EACH NUMBER	1.5	Every number will be shown 1.5 seconds
PAUSE BETWEEN NUMBERS	0.5	The screen will be blank (= black) for 0.5 second between each number.
PAUSE BETWEEN GAMES	2.0	A short 2 second pause will be held before next game will start.
NR OF SLIDES	50	50 numbers will be shown regardless of level chosen.
PROBABILITY TO SEE PREVIOUS NUMBER?	25 %	E.g., in level 1, there is every time a 25 % probability that the previously shown number will be repeated.
SHOW SCORES AFTER EACH SESSION?	YES	Final scores will be shown after a session of 7 games have been played. Scores are never shown between games.
SHOW PICTURES FROM PICTURE FOLDER?	NO	No pictures will be displayed, only numbers

5.3 Machine learning process

The steps done in this experiment related to machine learning were:

- EEG-data for all three participants in the EEG-experiment was augmented with the levels (numbers 0 – 2) being played for every measurement. For the “deep breathing” or relaxation pauses before the different levels, the letter B was augmented.
    - To be able to synchronize the game data with the EEG-signals, it was crucial that the timing for both was synchronized, this was solved by using same computer for the n-back game and for the data collection software.

  - As this was a small experiment, it was found easiest to use Excel-functions for automatically synchronizing game and EEG-data and augmenting the levels played.
- The data was divided into CSV-files of 4 seconds each, resulting in about 260 files for each participant. To perform this data split, the researcher had created a short and versatile Python-program.
- EEG-data is relatively unique and only on a broad level comparable between different persons. Due to this, and because of varying data quality between the participants and the recorded channels, it was decided to not try to combine data from all participants into one dataset. Instead, three separate projects were created in the Edge Impulse platform, and the channels with highest data quality were chosen for each project.
- While Edge Impulse is not a full-fledge data processing or signal analysis platform, it *can* be used to preprocess data. With previous experience from similar experiments, the researcher decided to use a preprocess step named *Spectral Features*. This is basically filtering the raw and sometimes noisy data, and extracting features i.e., significant components from the data. It is also significantly reducing the final dataset size, in these projects from 1024 data points to only 19 for each 4 second sample, an over 50-fold reduction!
- Discovering what neural network parameters to use for a certain application and dataset is according to Sukthomya (2005) often based on experience, trial and error. In this experiment, the size of the input (19 neurons) and output layers (2 neurons) were determined by the previous steps. To find the optimal number of hidden layers, and the optimal number of neurons in each of them, following permutations were tried empirically, noticing the performance and accuracy of each of them:
  - 2 hidden layers with 20 respective 10 neurons (default in Edge Impulse)
  - 2 hidden layers with 30 respective 15 neurons
  - 2 hidden layers with 10 respective 5 neurons
  - 2 hidden layers with 15 respective 5 neurons
  - 2 hidden layers with 14 respective 4 neurons
  - 3 hidden layers with 30, 20, and 10 neurons
  - 1 hidden layer with 30 neurons

- Of these combinations, 2 hidden layers with 14 respective 4 neurons was found to be optimal for this scenario and dataset. A larger dataset is expected to need somewhat different parameters.

6 RESULTS

6.1 N-back game

As the main deliverable of this study is a versatile n-back game which is particularly useful for researching about working memory, cognitive workload etc., with or without simultaneously measuring EEG-signals. For a Python developer, the game is furthermore easily extendable as the code is open-source (made freely available and may be redistributed and modified), heavily commented and documented (see Appendix 1 and 2), and new settings can be added with just one line of code. One positive side effect, due to the way the game was developed, is that it can be localized to other languages based on the Latin alphabet by simply editing a text file, in this case without changing a single line of code.

6.1.1 Iterations

This project consisted of a total of five iterations, numbered from 0 to 4. The first iteration consisted of a prototype that as such worked, but with limited functionality and features, hence it was numbered as iteration 0 to indicate it should not be used for real purposes, only for testing. Iterations 1, 2, and 3 were versions which could be tested and used in real scenarios. Finally, iteration 4 consisted only of activities securing the game could be used, maintained, and developed further by other persons.

Table 5 lists all iterations and functionality added or changed in each of them. Though very minor changes like e.g., renaming menus or other text have not been included in this list, they have all been communicated to the client. In addition, a lot of code

changes have for every iteration been done “behind the scenes” but only those of importance to the client or users are included in the list.

Table 5. Iterations and features

ITERATION	FEATURE
0 PROTOTYPE	GUI (Graphical User Interface), including main menu and settings menu, only possible to use mouse, not keyboard yet
	Game modes: • single game • multi game = several single games after each other
	Settings for: user ID, game level (1 – 4), time related, how many numbers to show for each level, screen resolution, font sizes, log file name
	Developed functionality so settings can **easily and flexibly** be added or changed in future iterations. • **Adding** a completely new setting only needs editing a text file and one line of code to read the new setting. • **Changing the name** of an existing setting, or how many alphanumeric characters are expected as maximum, is done by editing the same text file. This also enables **translating the game to other languages** without changing any code at all**!**
1 FIRST USABLE VERSION	Possibility to use both mouse and keyboard (Space = left mouse, Enter = right mouse)
	Scoring system
	Possibility to require action (Enter/right mouse) when shown number **is not** same as previous n number. No impact on scoring yet.
	Log file in CSV-format, data logged is person ID, timestamp when number was shown, timestamp when user clicked, button clicked, clicked correctly or not, number “queue” (5 last numbers shown)
	Setting for probability-% to see previously shown number (as per the n-level), a recommended setting is 25 %. Thru this setting, the shown numbers are more likely to reoccur compared to a lower %.
	Lots of code changes “behind the scenes” to ensure new functionality can be added purely by adding new Python modules. This will also enable unit testing i.e., the whole software does not need to be tested for every new feature, it is typically enough to only test the new feature and what happens right before and after it.
2	Added game mode to play random multi games i.e., several games after each other where the level is randomly chosen. The user only needs to decide how many levels to play, and which levels e.g., levels 0 – 2, 1 – 3, etc.
	Scoring system is enhanced to also include the option where the user should react if the number shown **is not** same as previous
	Logging more information to make the log file easier to understand
	Added sound when “deep breathing” time has ended, and the user can start playing the game.
	New settings added for random multi game mode

	The user can undo changes done to the settings, and also revert the changes to "factory" defaults.
	The game can be stopped at any time with ESC-button.
3 FINAL VERSION	Level 0 added. Level 0 means that the user should react when a target number is shown. Added a setting for the target number, default number set to 4.
	Added a setting for which sound file to use when breathing time has ended. This makes it easy for the user to choose a personalized sound. Found a few royalty free sounds that were added to the repository.
	Added a setting to decide if scores should be shown or not after a session. In multi game session scores are never shown, only after the session has ended, depending on the new setting.
	Added functionality to show pictures • Setting to choose if pictures should be shown or not • Setting for a picture folder, ONLY pictures should be in that folder. • Settings for picture size (x and y), and picture position (x and y). • A randomly selected picture from the folder will be shown with the selected size and at the selected position, **simultaneously** with the number. The pictures are automatically scaled to fit the given size; hence they should have same aspect ratio, otherwise they will be stretched and might look strange.
4	• No code changes. • Created a Github repository (https://github.com/baljo/n-back) and uploaded the game to it. • Created instructions for how to install the game, how to use the game, and how to add settings as a developer. Instructions uploaded to the Github repository.

6.1.2 The final product

The final product consists of a main menu as seen in Figure 21, and in Figure 22 an example of a number shown, together with a picture intended to evoke emotions. The different settings available are shown in Figure 23, they are further described in Appendix 1.

Figure 21. N-back game, main menu

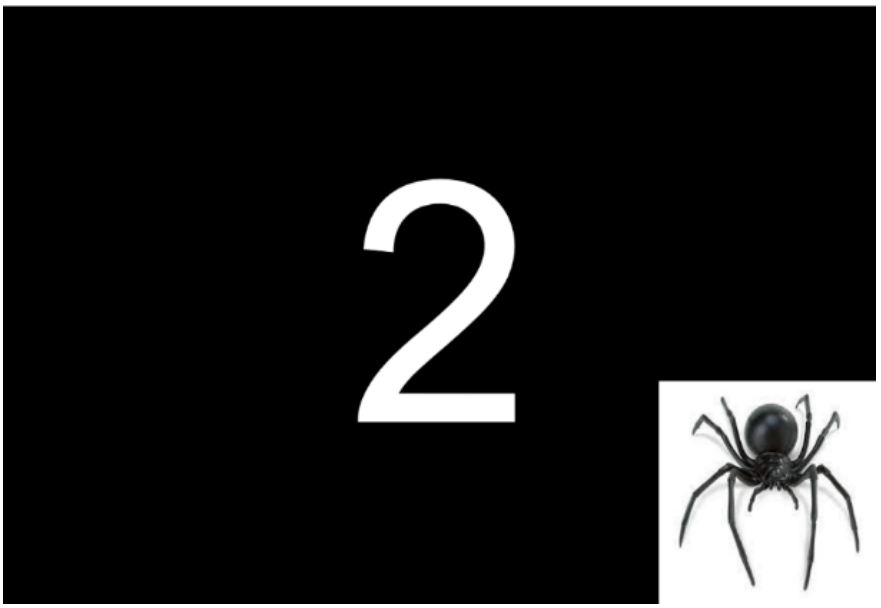


Figure 22. Game play window, spider shown to evoke emotions

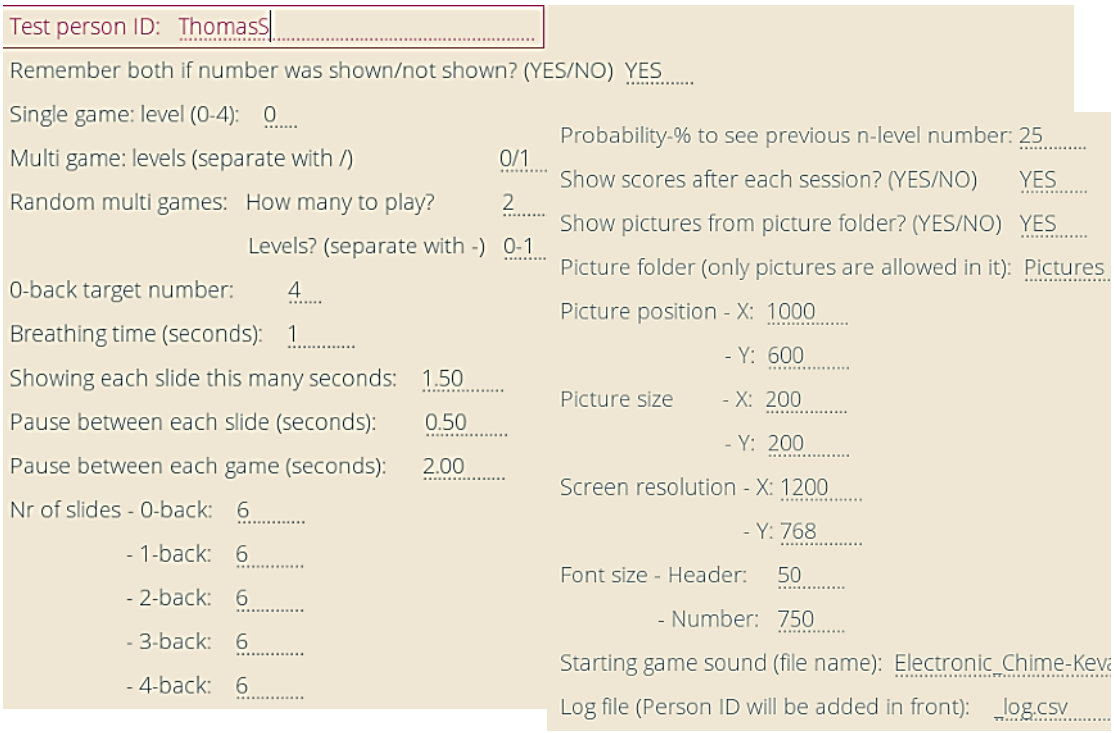


Figure 23. Available settings

6.1.3 Game feedback survey

The results of the short questionnaire sent to the client are listed in Table 6 and Table 7.

Table 6. Quantitative questions

QUESTION	RESPONSE
ON A SCALE 1 – 5 (1 = VERY UNSATISFIED, 5 = VERY SATISFIED)	
• How well did the n-back game fulfil the requirements and expectations?	5
• How easy is it to set up the game for different experiments, e.g. by changing the settings?	5
ON A SCALE 1 – 10 (1 = VERY UNLIKELY, 10 = VERY LIKELY)	

• How likely would you recommend the game to other researchers for similar experiments?	9

Table 7. Qualitative questions

QUESTION	RESPONSE
WHAT COULD BE IMPROVED?	At this stage it is difficult to mention any improvements as, in fact, all the changes that were proposed, were also implemented. Ideas for improvement may come during experiments. A problem with screen resolutions when using extended displays came up, so maybe this is something that needs to be addressed. The game allows different settings and modes (single game, predefined sequence of games, random sequence of games) but a user who is not familiar with the game may get confused. Probably the thesis will provide a kind of manual.
WHAT WORKS WELL?	The variety of settings and the possibility to define the position and size of the images!
ANY OTHER COMMENTS?	None

As a summary of the survey, it shows that the client was very satisfied with the n-back game and its features, and how flexible its is because of the various settings available.

6.2 EEG-experiment

Following chapters present the results from the EEG-measurements done during the experiment. Chapter 6.2.1 describes the spectrograms produced from the data, chapter 6.2.3 describes heart related data, and finally chapter 6.2.4 shows the results of using machine learning to classify EEG-data.

It was beforehand known that the results could not be deemed statistically significant as only three persons would participate in the experiment, and they would only participate once. Nevertheless, the purpose was to see if even this limited data might give some indications if different cognitive workload can be assessed with a consumer-based EEG-device. A full study using the developed n-game is planned to be done later.

After analyzing the data, it was noticed that some channels had very noisy or unclean data for some of the participants. The reasons for this are not certain, but it might be that the headband was not tight enough, so the electrodes did not make very good contact with the skin. Another reason might have been that the participant moved the head, although this was not noticed by the observer.

6.2.1 Game play and observations

Quantitative data

Table 8 shows recorded results from all three participants. As the game levels were randomly chosen, the overall degree of difficulty could have been quite different between the three participants. However, the levels chosen were in the end rather similar for all.

Table 8. Experiment session results

PERSON	LEVELS PLAYED	FINAL SCORE (of maximum 342)
1	2, 1, 0, 1, 1, 1, 2	340
2	2, 0, 1, 0, 1, 2, 2	340
3	2, 0, 2, 2, 0, 2, 0	337

Qualitative data

After each participant had completed the 15-minute session, the participant was asked to verbally provide spontaneous comments about the session. These as well as comments from the observer are found in Table 9.

Table 9. Comments from test persons and observer

PERSON	OWN COMMENT	OBSERVER COMMENT
1	…I stayed quite focused 'in the zone'… …my strategy for level 2 was to think that 'if next number is <n>, I need to click'…	Seemed to have no struggles, and the only two errors occurred at level 2.
2	…I started to feel a bit fatigued…	The participant's mobile phone started to vibrate during the session at level 1. When asked if this had an impact, the answer was "I noticed it, but it did not distract." The only 2 errors occurred in the final game at level 2.
3	…[when seeing the final score] I thought it went worse…	Signs of frustration after having to play at level 2 twice in sequence. All errors occurred at level 2.

After the session, the participants were asked for possible changes or improvements to the n-back game. The objective with this was to find out the usability and user-friendliness of the game.

One recommendation given was to remove the final score at the end of the session. The reason for this request was that the participant thought it might be good to not know the score if same person is playing several sessions after each other, as it might impact on the performance. In this experiment this did not matter as each participant played all the games after each other. However, an option for showing or not showing scores was actually already implemented in the settings, so it is up to the researcher to decide which option to use. In this session the final score was shown.

Another recommendation was to raise the probability for higher levels to occur in a random multi game. Now each level can occur with same probability, and as levels 0 – 2 were selected as available, each level had 1/3 probability to occur. The reason for this recommendation was that levels 0 and 1 are very easy to play and there is a chance that the harder level 2 is selected only once, or not at all. This matter had also been considered but was not implemented in the game as it was not seen as a major issue. However, by using another game mode "Play multi games", and manually selecting e.g., levels 0, 1, 2, 2, 2, 1 and 2, the researcher can in a semirandom fashion circumvent this potential issue through controlling which levels are played, and in which order.

6.2.2 Spectrograms

Spectrograms are a type of graph where the data is divided into frequency bands over time. As an example, Figure 24 shows data from participant 1 and channel 2, the vertical y-axis indicates the frequency in Hertz (oscillations per second), and the horizontal x-axis is the timeline in seconds from the start of capturing EEG-data. The yellow color indicates the dominating frequency, which in the beginning was slightly below 5 Hz. For this participant, data was collected for 1035 seconds (17 minutes and 15 seconds), but due to technical issues in the beginning, the participant started the game first after approximately 100 seconds. This figure is also an example of where the data was of poor quality, hence it is also almost impossible to find e.g., when a level was started and when the participant was relaxing in between the games. Worth to notice is however that this spectrogram is created with raw data, without removal of noise or other artifacts.

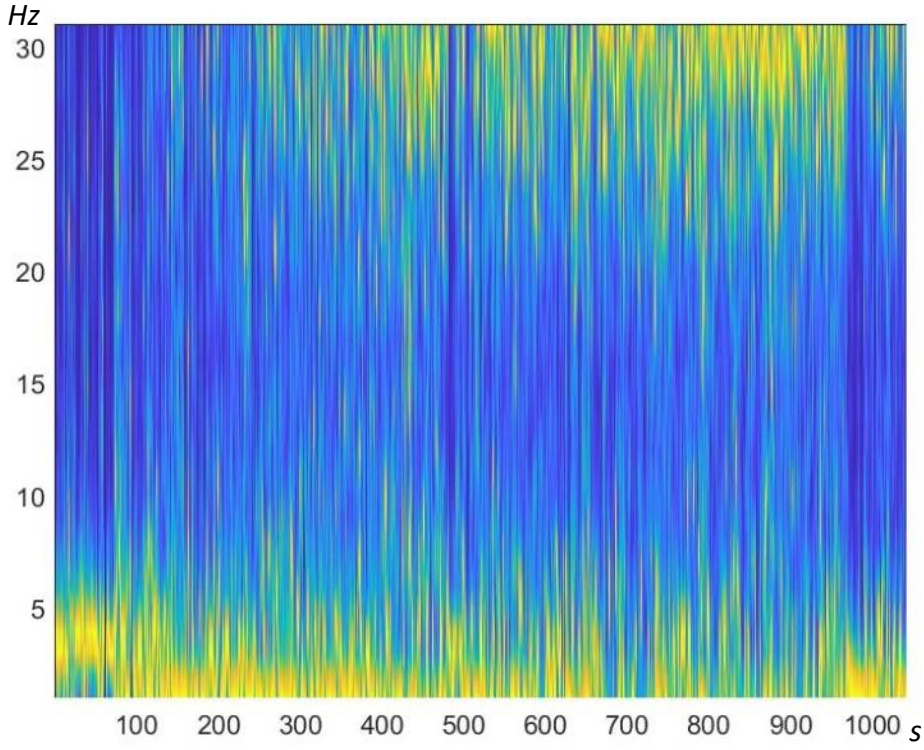


Figure 24. Spectrogram from participant 1, channel 2

Another example when the data quality was very good is illustrated in Figure 25. The relaxation periods before the levels are clearly visible as are the levels themselves. The annotated white numbers indicate which level was played. While it is not possible to draw any clear conclusions from this spectrogram, one might speculate that there might be some differences in the EEG-signals between level 2 and levels 0 and 1, where it seems that the frequency for levels 2 is a bit more consistent. Also this spectrogram is based on raw data.

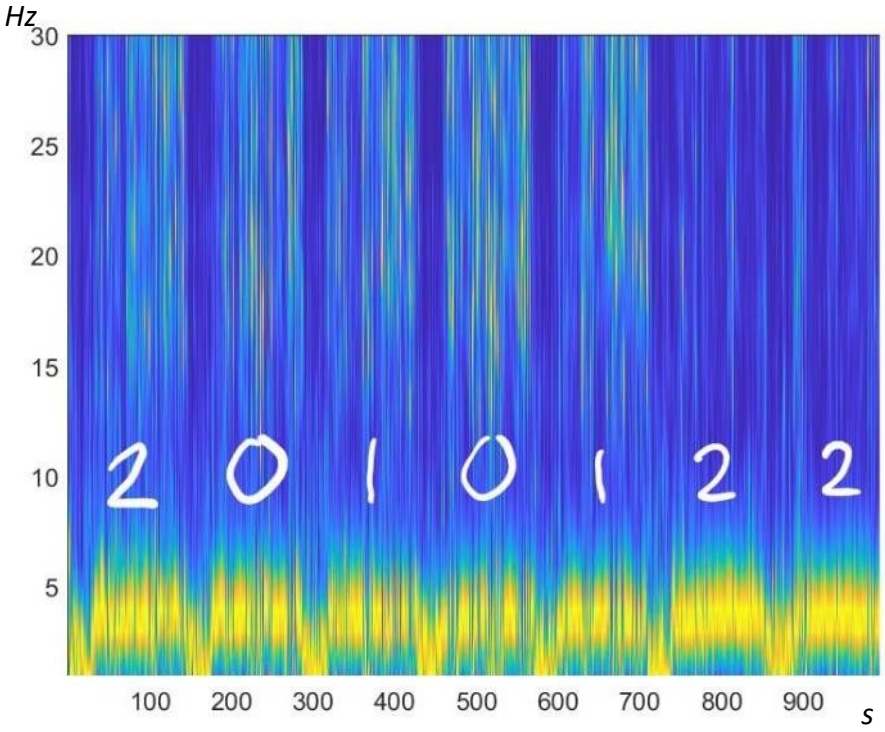


Figure 25. Spectrogram from participant 2, channel 2

Finally, an investigation of the data from all participants revealed that the data for participant 3 had least amount of interference and artifacts stemming from e.g. eyeblinks. Hence the research team chose to further analyze and clean this data, resulting in further spectrograms, one of them seen in Figure 26. Also in this spectrogram, the relaxation periods before the different levels are clearly visible, but here it is not possible to infer differences between the levels played.

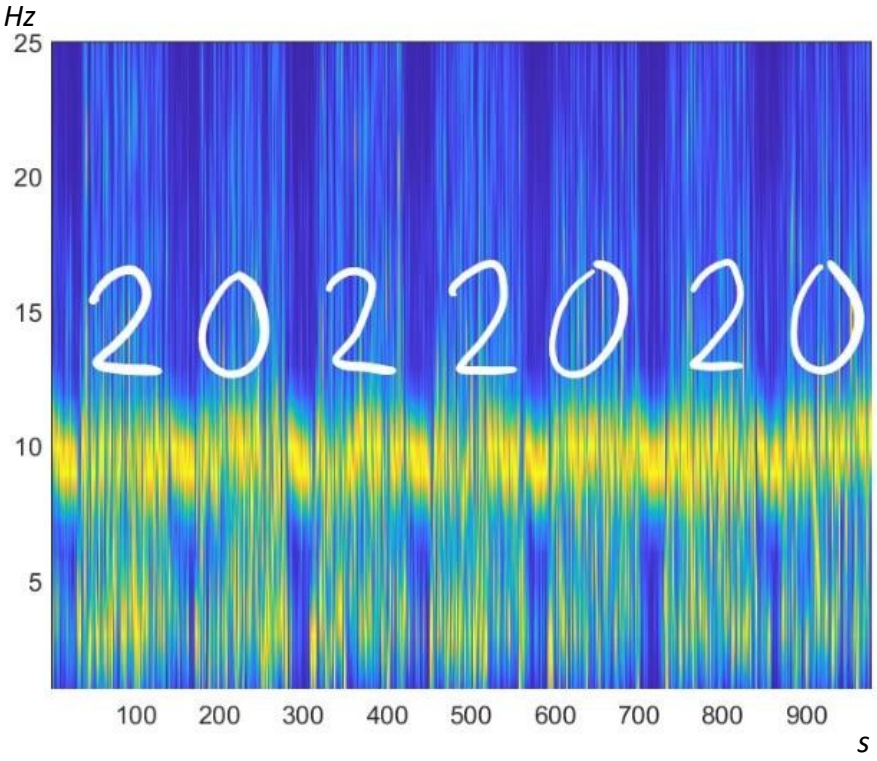


Figure 26. Spectrogram from participant 3, channel 1

This analysis was as such out of scope for this study, but the steps that were taken by the research team were described like this:

This is calculated over 2 sec windows; the center time of the first window is 1 s from the beginning of the recording and the following columns come in 1 s steps. Thus, the windows overlap by 1 s. The individual spectra (i.e., columns) are scaled so that maximum corresponds to value 1. About 9-10 Hz component well seen thought the recordings but the signal becomes more rhythmic during relaxation period.

The calculation is as follows:

- *Signals are filtered using a bandpass filter with the passband from 2…25 Hz using a linear phase filter*
- *Eyeblink artifacts are attenuated using wavelet transform (as there were not many eyeblinks in the recording, this is not critical)*

- *Spectrogram is calculated using the periodogram method (2 sec segments with mean value removed, overlap 1 sec, hamming window, subsegment length 0.5 sec (125 samples) with 0.25 sec (62 samples) overlap)*
- *Individual spectra are scaled so that maximum value is corresponding to 1*

6.2.3 Heart related data

As the Muse EEG-device also includes a PPG-sensor (photoplethysmography), it is possible to collect heart related data from it, and with proper tools even calculate the heart rate. PPG-sensors work by emitting green or red light from a LED (Light Emitting Diode) on the skin and measuring the light reflection. This reflection data can be used as such, or further be processed to determine heart rate by measuring how much blood flows through the illuminated spot as the heart beats.

For analyzing and processing the collected PPG-data, HeartPy, an open-source Python-program developed for this purpose, was used. The program however revealed that the data quality was not good enough for calculating heart rate, without considerable cleaning effort needed to be done. As heart rate was not in the main scope for this study, it was decided to be left out. Nevertheless, the raw PPG-data collected from participant 3 is shown in Figure 27, the blue line and its peaks correspond to heart

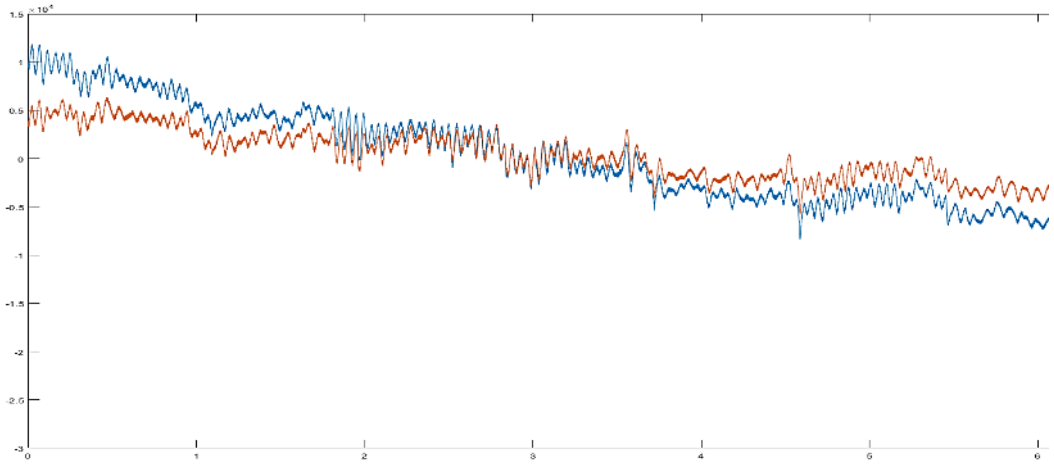


Figure 27. Raw PPG-data

rate, and the red line and its peaks to respiration. When zoomed in, like in Figure 28, the peaks are easier to distinguish.

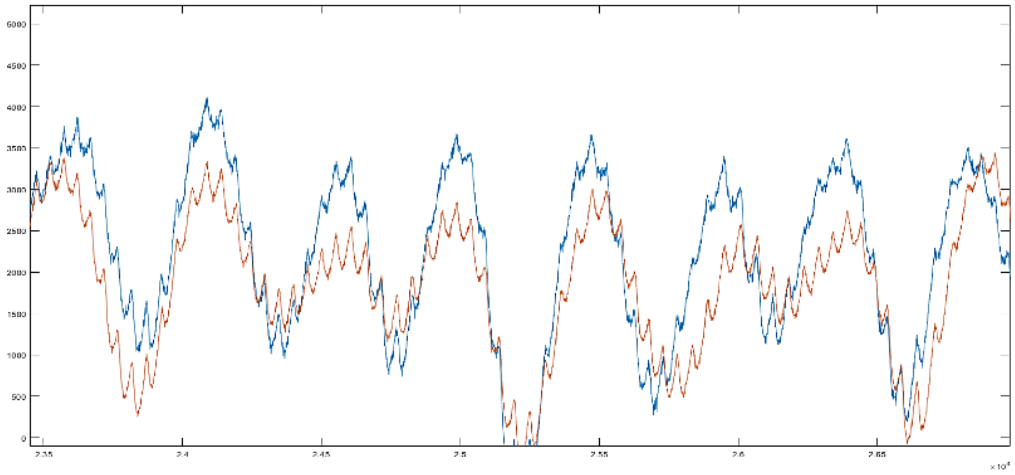


Figure 28. Zoomed in PPG-data

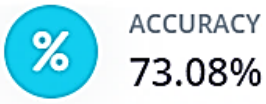
6.2.4 Classification of n-game levels with machine learning

After training in Edge Impulse using the data from the EEG-experiment, the platform reported the calculated accuracy, a confusion matrix, and F1-scores, all of them revealing the training performance. See Figure 29 for an example where the algorithm was able to distinguish between level 1 and the relaxation period with 85.7 % accuracy. This accuracy is surprisingly high considering the size and quality of the dataset.

	1	B
1	91.7%	8.3%
B	17.4%	82.6%
F1 SCORE	0.81	0.88

Figure 29. Training performance from participant 2, channels 1 and 3

A machine learning algorithm only performing well on training data, is however of no or very limited use. To find out the real performance, the model needs to be tested on data it has *never seen before*. For this purpose, the Edge Impulse platform is by default only using 80 % of the data for training and saving 20 % for model testing. Quite often, especially when the dataset is small, or if the neural network has been over trained, the model performs worse in the testing than in the training phase, sometimes even so poorly so it is of no use at all. As Figure 30 shows, the result from the model testing in this case was expectedly lower, but still much higher than a random model is expected to give.



	1	B	UNCERTAIN
1	82.1%	10.7%	7.1%
B	33.3%	62.5%	4.2%
F1 SCORE	0.78	0.71	

Figure 30. Model testing performance, participant 2, channels 2 and 3

A summary of the results from all three participants is listed in Table 10, where the column *Channels* refers to which channel (= electrode) had usable data, and column *Levels* refers to the levels played (or B for relaxation period).

Table 10. Classification training and testing results

PARTICIPANT	CHANNELS	LEVELS	TRAINING ACCURACY	TESTING ACCURACY
1	2	1, 2	83.3 %	51.6 %
2	1, 3	1, B	85.7 %	73.1 %
2	2	2, B	79.4 %	76.3 %
3	1, 3	1, B	77.8 %	60.0 %

While one should be cautious to draw any definite conclusions from these results, they at least seem to confirm that, similarly as is visible in the spectrograms, it is possible to distinguish between EEG-data from participants 2 and 3 playing a level in the n-back game, and the same persons relaxing before playing. Although the training accuracy for participant 1 when playing levels 1 and 2 is relatively high, the model testing accuracy is quite low and not significant enough to clearly distinguish the levels from each other.

7 CONCLUSIONS AND DISCUSSION

Following sub-chapters discuss how the objective of the study was reached, success contributors, limitations as well as recommendations identified, and contributions to research.

7.1 How has the purpose been reached

The purpose of this study was to increase knowledge about how cognitive workload can be assessed through analyzing data coming from a consumer-based EEG-device. To reach this objective, the first aim was to program a n-back memory game. This

game was to be used to fulfil the second aim: preliminary testing of the concept to assess the potential of the technology. The third aim was to, based on the above-mentioned test, define important aspects when collecting data from EEG-devices.

Furthermore, the research questions (RQ) set for this study were:

- RQ 1: How can cognitive workload be assessed with a consumer-grade EEG-device?
  - RQ 1a: What tools are needed?
  - RQ 1b: How can the tools be used?

Earlier research show that it is indeed possible to assess cognitive workload with *professional* devices measuring EEG-signals from the brain. What has though not been exhaustively researched is using a *consumer-based* EEG-device for the same purpose, which is what this study tried to tackle.

As response to RQ 1a – What tools are needed – following has been identified:

- A *consumer-grade EEG-device* is obviously needed to measure brain signals, in this case the project research team had purchased a Muse S EEG-device that was used.
- Tools for streaming or recording EEG-data from the device to a computer. In this study the Petal Metrics-software was used.
- Tools to analyze, interpret, classify, and visualize data from the EEG-devices, for example Matlab, R, Python, Excel, Edge Impulse, etc. In this study Excel and Python was used to prepare data for machine learning classification in Edge Impulse.
- A *method of inducing increased cognitive workload* is required. This can be a completely analogue method e.g., words or numbers printed on paper and to be remembered, or a digital method like a computer or mobile phone program. After evaluating existing solutions, it was decided to design and develop an open-source *n-back game* that would be so versatile so it could be used for a small experiment, but also for later experiments.

For RQ 1b – How can the tools be used – following has been identified:

- For securing proper data quality, the EEG-device needs to be set up properly and tested before any experiment. Likewise, data connectivity through Bluetooth or other technologies, needs to be secured as well as possible.
- The physical test environment needs to be free from disturbance.
- When the n-back game developed as part of this study is used, it should be configured so that the levels to be played are consistent for every participant.

7.1.1 Main success contributors

Following advantageous choices and success contributors have been identified:

- After evaluating existing solutions for the n-back test, it was decided to program it from scratch in the Python programming language, *using an iterative and incremental method.*
    - As this development project had rather limited scope, almost any software development method could have been used to ultimately fulfil the requirements. Main differences would have been the overall time needed, and how many iterations were required. Still, the chosen method was found to be ideal as it enabled a customer-oriented and fast-paced project.
- The n-back memory game was by the researcher programmed according to functional and non-functional requirements.
    - Defining both functional as well as non-functional requirements ensured that the program works as expected when delivered (functional requirements), and also is easy and logical to use, flexible and maintainable (non-functional requirements).
    - Code modularization ensured less bugs and made the testing easier and faster.
- The game was used in a small experiment with only three participants wearing a consumer-based EEG-device.

  - Data was collected through Bluetooth-connection to a computer running the Petal Metrics-software, streaming the data into CSV-files for later analysis. Possible time synchronization issues were mitigated by using same computer for both gameplay and EEG-data collection.
  - The experiment was observed by the author to identify what is working well, and what could be improved. No major improvement areas were identified for the n-back game itself.
  - A survey answered by the research team shows they were very satisfied with the n-back game and how it can be used for similar future experiments.
- The results are not statistically significant, however:
  - Some of the spectrograms clearly show that the EEG-signals are different during periods of rest, and during periods of playing the game. Speculatively, the exact level played might also be visible for one of the participants.
  - When running the same data through a machine learning platform, the results seem to confirm what the spectrograms indicated i.e., EEG-signals can be classified according to periods of rest or periods of more intense cognitive work.

7.1.2 Conclusion

To summarize, the objectives of this study have been fulfilled, especially when it comes to the n-back game developed which was found flexible for experimenting with cognitive workload, and also easily extendable for developers. While the data collected from the small EEG-experiment was of varying – and sometimes disappointing – quality, it could still be used in triangulating and achieving the purpose of this study, using two separate tools: spectrograms, and classification through machine learning. Especially reflecting the size and quality of the collected EEG-data, also this part of the study can be considered successful.

7.2 Limitations of this study

Limitations identified as part of this study are described further in the following sub-chapters. General recommendations, and those related to the limitations, are found in chapter 7.3.

7.2.1 Quality of the process used in the study

Although the objectives and aims of this study are considered fulfilled, some potential gaps and subsequent improvement areas have been identified related to *how* the study was done. These areas are:

- Exploratory literature search related to EEG-data: In short this was done by mainly using a combination of the words in the search criteria "analyze EEG data cognitive workload consumer-based". As mentioned in chapter 3.5, the researcher had not access to all the material found, but by having wider access or purchasing interesting articles, more applicable literature might be found. In addition, searching for stress or working memory, instead of cognitive workload, might result in similar but not necessarily completely applicable literature.
- Software development process: Once development and subsequent testing had started, only the researcher and the client tested the intermediate versions. As the software complexity grew almost exponentially during each iteration, the client could have been emphasized about what to specifically test and how to test it. This was not done in the first iterations. Moreover, having only one end user testing – in addition to the developer – is not ideal.
- EEG-experiment session: Due to tight schedule, there was not enough time available to do another round of experiments, or to investigate connection issues and possible reasons for them. This hampered at least the data quantity collected, perhaps also the data quality.

7.2.2 Data quantity and quality limitations

The experiment done as part of this study was by design very small having only three participants, and the results are not statistically significant.

Data collected from the consumer-grade EEG-device during the experiment was of varying quality, and the full potential of this study might not have been reached. While the reasons for the varying data quality were not further investigated, the researcher speculate that connectivity issues might have played a role. These issues might have been due to loss of Bluetooth data packets, and sometimes poor contact between EEG-sensors and the skin of the participants. Other possible reasons are environmental related, like closeness to fluorescent lamps or 50 Hz artefacts from power outlets. Experiments done by the researcher outside of this study resulted in better quality, albeit this was with an older version of the EEG-device than being used in the experiment, and in another environment.

Furthermore, heart rate and respiration data collected by the EEG-device's PPG-sensor could not be exploited due to data quality issues.

The main focus of consumer-grade EEG-devices is on other areas than pure EEG-research – like meditation and relaxation – and usability as well as user experience have probably higher priority than technical data quality and consistency. Thus, it is possible that to be able to successfully use these devices in research, special attention needs to be put on preventing poor connectivity, on minimizing possible interference from fluorescent lamps and power outlets, and also on data cleaning.

7.2.3 Limited features in the n-back game

The n-back game was by the researcher developed from scratch for one purpose only: to induce increased cognitive workload which might or might not be assessed with for example analyzing EEG-data. The game has many settings, it is flexible and easy to set up as well as to use, and it is for a Python programmer easy to extend. Despite this,

it cannot compete with full-fledged psychological test platforms which typically include many more functions and features apart from n-back.

The game is designed to be completely stand-alone i.e., it is not itself receiving data from EEG-devices. If EEG-data is required, external streaming software or solutions are needed, which increases the effort needed to synchronize EEG-data with game data. Also, the more components involved, the higher the risk is that technical or data issues occur.

While the n-back game has been verified to work as designed and expected to in normal situations, it has not been tested on other than Windows 10-computers, and there is limited error-control. This might cause issues when running the game on Linux- or Mac-computers, using extended monitors, deleting, or not installing all files needed, etc.

7.3 Recommendations

Several recommendations for improving the n-back game itself and for further research have been identified and they are unfolded in the following subchapters.

7.3.1 N-back game

Recommendations for the n-back game are based on feedback from the research team, from participants in the experiment, and the researcher's own:

- Error control is strongly recommended to implement
  - Due to limited time available for the development, no specific error control using Python functionality is built in, although it had been considered. The program might "crash" if the settings file is missing or corrupted, or if other unexpected issues occur.
  - Investigate why the game does not support extended monitors.
- Implement probability setting for higher levels to occur in multi random games.

  - If playing only a few random games, “too many” low-level or high-level games might have been randomly chosen. By adding a new probability setting for high levels to occur, this possible issue could be addressed.
- Integrate n-back game and EEG-streaming
  - There are a few Python-libraries available for streaming EEG-data to a computer, and by integrating this functionality into the n-back game there would not be any need to manually consolidate data files, reducing the risk of manual errors. Furthermore, this would also streamline and simplify the research process. Worth noticing however, is that this integration is only applicable for devices using the same connection standard, so it should be optional to use.

7.3.2 Recommendations for further research

- Compare consumer-based devices with professional devices
  - A wider systematic review of consumer-based and professional EEG-devices is recommended to find out strengths and weaknesses of both when assessing cognitive workload.
  - It is recommended to use the n-back game or similar tools, and with same participants using both type of devices.
- Search for literature related to assessment of stress or working memory with EEG-devices
  - This might reveal research similar to assessment of cognitive workload.
- Investigate data quality
  - As was found from the experiment done in this study, the data quality was varying. More research would be needed to find out why this was the case, and how the quality might be improved.
- Research if and how heart rate and respiration data from PPG-sensors add value
- Wider study needed with more participants playing longer games, with harder and harder levels
  - The experiment done as part of this study was by design very small, and the results are not statistically significant, also the levels played by the three participants were not identical.

- To possibly more clearly see if higher cognitive workload is visible in EEG-signals, it is recommended to set up a game session for each participant where the difficulty is constantly increasing e.g., levels 0, 0, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3 etc.

7.4 Contributions to research

The main contributions to research from this study is a flexible, configurable, and extendable n-back game that has been proven to be very useful for inducing cognitive workload, as well as promising results showing it is possible to also use consumer-grade EEG-devices for assessing this workload.

REFERENCES

Acharya, J. N., Hani, A., Cheek, J., Thirumala, P., & Tsuchida, T. N. (2016). American Clinical Neurophysiology Society Guideline 2: Guidelines for Standard Electrode Position Nomenclature. *Journal of Clinical Neurophysiology : Official Publication of the American Electroencephalographic Society*, *33*(4), 308–311. https://doi.org/10.1097/WNP.0000000000000316

Almogbel, M. A., Dang, A. H., & Kameyama, W. (2019). Cognitive Workload Detection from Raw EEG-Signals of Vehicle Driver using Deep Learning. *International Conference on Advanced Communication Technology, ICACT*, *2019-February*. https://doi.org/10.23919/ICACT.2019.8702048

Al-shargie, F. M., Tang, T. B., Badruddin, N., & Kiguchi, M. (2016). Mental Stress Quantification Using EEG Signals. In F. Ibrahim, J. Usman, M. S. Mohktar, & M. Y. Ahmad (Eds.), *International Conference for Innovation in Biomedical Engineering and Life Sciences* (pp. 15–19). Springer Singapore.

American Psychological Association. (2022). *Cognitive load*. https://dictionary.apa.org/cognitive-load

Cowan, N. (2010). The Magical Mystery Four: How Is Working Memory Capacity Limited, and Why? *Current Directions in Psychological Science*, *19*(1), 51–57. https://doi.org/10.1177/0963721409359277

Dabbagh, M., & Lee, S. P. (2014). An approach for integrating the prioritization of functional and nonfunctional requirements. *The Scientific World Journal*, *2014*. https://doi.org/10.1155/2014/737626

Despa, M. L. (2014). Comparative study on software development methodologies Comparative study on software development methodologies. *Database Systems Journal*, *3*.

Farnsworth, B. (2021). *What is EEG (Electroencephalography) and How Does it Work?* https://imotions.com/blog/what-is-eeg/

Frost, A., Moussaoui, S., Kaur, J., Aziz, S., Fukuda, K., & Niemeier, M. (2021). Is the n-back task a measure of unstructured working memory capacity? Towards understanding its connection to other working memory tasks. *Acta Psychologica*, *219*, 103398. https://doi.org/10.1016/J.ACTPSY.2021.103398

Hervé, A. (1994). A NEURAL NETWORK PRIMER. *Journal of Biological Systems*, *2*, 247–283.

Hinrichs, H., Scholz, M., Baum, A. K., Kam, J. W. Y., Knight, R. T., & Heinze, H. J. (2020). Comparison between a wireless dry electrode EEG system with a conventional wired wet electrode EEG system for clinical applications. *Scientific Reports*, *10*(1). https://doi.org/10.1038/S41598-020-62154-0

Hou, X., Liu, Y., Sourina, O., Tan, Y. R. E., Wang, L., & Mueller-Wittig, W. (2015). EEG Based Stress Monitoring. *2015 IEEE International Conference on Systems, Man, and Cybernetics*, 3110–3115. https://doi.org/10.1109/SMC.2015.540

Interaxon. (2022a). *Lifestyle - Google Drive*. https://drive.google.com/drive/folders/1TI0iBTaXHbJo6-X0qi8W47gYNVTKK2Hk

Interaxon. (2022b). *Product - Google Drive*. https://drive.google.com/drive/folders/1-L1khsOuV7cOy_YolD_ljbGm0szIdT1F

Jiantao, P. (1999). Software Testing. *Dependable Embedded Systems*.

Jun, G., & Smitha, K. G. (2016). EEG based stress level identification. *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 3270–3274. https://doi.org/10.1109/SMC.2016.7844738

Katmah, R., Al-Shargie, F., Tariq, U., Babiloni, F., Al-Mughairbi, F., & Al-Nashash, H. (2021). A Review on Mental Stress Assessment Methods Using EEG Signals. *Sensors*, *21*. https://doi.org/10.3390/s21155043

Kirchner, W. K. (1958). Age differences in short-term retention of rapidly changing information. *Journal of Experimental Psychology*, *55*(4), 352–358. https://doi.org/10.1037/H0043688

Kutafina, E., Heiligers, A., Popovic, R., Brenner, A., Hankammer, B., Jonas, S. M., Mathiak, K., & Zweerings, J. (2021). Tracking of mental workload with a mobile eeg sensor. *Sensors*, *21*(15). https://doi.org/10.3390/s21155205

Lau-Zhu, A., Lau, M. P. H., & McLoughlin, G. (2019). Mobile EEG in research on neurodevelopmental disorders: Opportunities and challenges. *Developmental Cognitive Neuroscience*, *36*, 100635. https://doi.org/10.1016/J.DCN.2019.100635

Lutz, M. (2013). *Learning Python*. 491–492. https://books.google.com/books/about/Learning_Python.html?id=4pgQfXQvekcC

Moren, J., & Shipulin, K. (2021). *Identifying Mental Workload in a gaming scenario using consumer-grade EEG*. https://doi.org/10.17045/sthlmuni.16900159

Neuroelectrics. (2022a). *Enobio® EEG systems | Neuroelectrics*. Neuroelectrics. https://www.neuroelectrics.com/solutions/enobio

Neuroelectrics. (2022b). *_Enobio32_5G_1.webp (250×250)*. https://www.neuroelectrics.com/img/product/_Enobio32_5G_1.webp

Niculescu, A., Cao, Y., & Nijholt, A. (2010). Manipulating stress and cognitive load in conversational interactions with a multimodal system for crisis management support. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *5967 LNCS*, 134–147. https://doi.org/10.1007/978-3-642-12397-9_11

PYPL. (2022). *PYPL PopularitY of Programming Language index*. https://pypl.github.io/PYPL.html

Reichheld, F. F. (2003). The One Number You Need to Grow. *Harvard Business Review*, *81*(12), 46–54. https://lillukka.samk.fi/login?url=https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip&db=buh&AN=11587407&site=ehost-live&scope=site

Ryu, K., & Myung, R. (2005). Evaluation of mental workload with a combined measure based on physiological indices during a dual task of tracking and mental arithmetic. *International Journal of Industrial Ergonomics*, *35*(11), 991–1009. https://doi.org/10.1016/J.ERGON.2005.04.005

Scrivener, C. L., & Reader, A. T. (2022). *Variability of EEG electrode positions and their underlying brain regions: visualizing gel artifacts from a simultaneous EEG-fMRI dataset*. https://doi.org/10.1002/brb3.2476

Soufineyestani, M., Dowling, D., & Khan, A. (2020). Electroencephalography (EEG) technology applications and available devices. In *Applied Sciences (Switzerland)* (Vol. 10, Issue 21). https://doi.org/10.3390/app10217453

Stanton, A. L., Stanton, J. M., Balzer, W. K., Smith, P. C., Parra, L. F., Mercer, W. M., Ironson, G., Sinar, E., Julian, A., Fischer, G., O'connor, M., Sandman, B., Moore-Hirschl, S., Paul, K.-R., Hayes, B., Schmieder, R., Hahn, S., Haff, M., Vandekemp, H., … Silverman, M. (2001). EDUCATIONAL AND PSYCHOLOGICAL MEASUREMENT A GENERAL MEASURE OF WORK

STRESS: THE STRESS IN GENERAL SCALE. In *Educational and Psychological Measurement* (Vol. 61, Issue 5).

Sukthomya, W. (2005). The optimisation of neural network parameters using Taguchi's design of experiments approach : an application in manufacturing process modelling. *Neural Computing & Applications*, *14*(4), 337–344.

Tampere University. (2022). *Projects | Intelligent Clothing | Tampere Universities*. https://research.tuni.fi/intelligent-clothing/projects/

Tiobe. (2022). *index | TIOBE - The Software Quality Company*. https://www.tiobe.com/tiobe-index/

Vance, A. (2022). *Brain Computer Interface Company Implants New Type of Device - Bloomberg*. Bloomberg. https://www.bloomberg.com/news/articles/2022-07-18/brain-computer-interface-company-implants-new-type-of-device

Wikimedia Commons. (2022). *International 10-20 system for EEG-MCN*. https://commons.wikimedia.org/wiki/File:International_10-20_system_for_EEG-MCN.png

Xu, K., Torgrimson, S. J., Torres, R., Lenartowicz, A., & Grammer, J. K. (2022). EEG Data Quality in Real-World Settings: Examining Neural Correlates of Attention in School-Aged Children. *Mind, Brain, and Education*. https://doi.org/10.1111/MBE.12314

APPENDIX 1 – GAME SETTINGS

All N-back game settings (slightly shortened from the game instructions at https://github.com/baljo/n-back#4-settings):

- **Test person ID** : unique ID for the test person, can be a name, initials, or something else
- **Remember both if number was shown/not shown?** : if YES, then you are expected to react both if the number shown was same (Space/Left mouse) or not same (Enter/Right mouse) as previous n. Only YES or NO should be stored here
- **Single game: level (0-4)** : Which level to play for the single game?
- **Multi game: levels (separate with /)** : When playing multi games, the levels will be in this order. Separate the levels with /.
- **Random multi games: How many to play?** : When playing random multi games, how many would you like to play?...
- **Levels?** : ...and which are the levels? Use e.g. 0-4, 1-2, or something else.
- **0-back target number**: When playing level 0, which is the target number the user should react upon?
- **Breathing time (seconds)**: Waiting time before the game starts, this is useful when simultaneously recording EEG-data to see if this relaxing period is visible or not in the data.
- **Showing each slide this many seconds**: A slide = a single number that will be shown. Change this to make the test harder or even more harder.
- **Pause between each slide (seconds)**: = Pause between each slide (seconds)
- **Pause between each game (seconds)**: Useful in multi game sessions to allow for more relaxation.
- **Nr of slides 0-back -> 4-back**: How many slides (= numbers) should be shown for each level?
- **Probability-% to see previous n-level number**: What is the probability that you will see the previous shown number? E.g., in level 1, and if the previous number shown was 5, probability is set to 25 %, then next number to be shown will be 5 with 25 % probability.
- **Show scores after each session?**: YES if you want to see the score, anything else here means NO.
- **Show pictures from picture folder?**: Do you want to see a smallish picture somewhere on the screen at the same time as a number is shown? YES or NO.
- **Picture folder (only pictures are allowed in it)**: If previous setting = YES, then this folder will be used to find pictures. Best is to have pictures of same aspect

ratio, otherwise they will be stretched. Only pictures are allowed in this folder, having other types of files there will cause issues!

- **Picture position - X + Y**: Where on the screen do you want this picture...
- **Picture size - X + Y**: ...and how large should it be?
- **Screen resolution - X + Y**: Depends on your screen, try e.g. with 1200 x 768.
- **Font size - Header**: This is the size of the text shown before a level starts.
- **Font size - Number**: Size of the numbers to be shown
- **Starting game sound (file name)**: This is the sound that is played after the breathing period, use a valid file name,
- **Log file (Person ID will be added in front):**: If your name is e.g. Thomas (or whatever is stored in Test person ID, the log file name will be Thomas_log.csv.

APPENDIX 2 – CODE SNIPPET

Figure 31 shows a snippet of the Python code and how it was commented, comments are in green color. Almost every row is commented describing what is done, and why. The whole program code with usage instructions is found at https://github.com/baljo/n-back.

```
n = [0, 0, 0, 0, 0]                                                    # number queue
game_score = 0


# ...and finally starting the main loop
for i in range(0, slides):                                             # the main loop
    n[4] = n[3]                                                        # moving numbers 1 step back, can be rewritten with...
    n[3] = n[2]                                                        # ...another for-loop, but this is easier to understand
    n[2] = n[1]
    n[1] = n[0]
    dice = random.randrange(0, 100 // prev_nr_probability + 1)         # % chance to see "previous" n-level number...
    if dice == 0:                                                      #
        n[0] = n[level]                                                # ... i.e. same as n-back (1...4)
    else:
        n[0] = random.randrange(0, 10)                                 # random number 0-9


    # level 0 needs special treatment
    if level == 0:

        dice = random.randrange(0, 100 // prev_nr_probability + 1)     # % chance to see the target number...
        if dice == 0:                                                  # if we "got lucky"...
            n[0] = _0_back_target                                      # the target number is what to show...
            expected = _0_back_target                                  # ...and thus the expected nr...
            expected_key = "Left/Space"                                # ...one of these is what to click
        else:
            n[0] = random.randrange(0, 10)                             # random number 0-9
            if n[0] != _0_back_target:
                expected = -1
                expected_key = "Right/Enter"                           # ...else one of these
            else:
                expected = _0_back_target                              # if we happened to get the target number by chance
                expected_key = "Left/Space"                            # ...one of these is what to click...

    # levels 1-4
    else:
        expected = n[level]                                            # expected number for levels 1-4
        if expected == n[0]:                                           # if shown number = expected...
```

Figure 31. Example of commented code.

APPENDIX 3 – GAME FEEDBACK SURVEY

On a likert scale 1 – 5, (1 = very unsatisfied, 5 = very satisfied):

- How well did the n-back game fulfil the requirements and expectations?
  - **Your answer (1-5)**:
- How easy is it to set up the game for different experiments, e.g. by changing the settings?
  - **Your answer (1-5):**

On a scale 1 – 10, (1 = very unlikely, 10 = very likely)?

- How likely would you recommend the game to other researchers for similar experiments?
  - **Your answer (1-10):**

Free text questions**:**

- What could be improved:
  - **Your answer:**
- What works well:
  - **Your answer:**
- Any other comments:
  - **Your answer:**