Personalized brain games for the elderly

reducing cognitive decline with AI-optimized memory training

Sanne Beijer
s.g.beijer@student.tue.nl
0941407

Nina Boelsums
n.m.boelsums@student.tue.nl
0964376

Pol Goetstouwers
p.goetstouwers@student.tue.nl
0904368

Jorrit van der Heide
j.v.d.heide@student.tue.nl
0942989

ABSTRACT

Rising numbers of the elderly population have caused an increase in reports of cognitive decline. Brain training systems are found to be effective to limit the negative effects of age on cognition. However, many brain training systems do not provide a personalized training experience, which is an important aspect of designing for users with declining cognition . This report describes the design of such a system. Making use of reinforcement learning, a brain training application was designed for reducing cognitive decline among the elderly. In the application, users are asked to remember faces. Based on the answer, the difficulty level of the game is adjusted to personalize the experience for its users. The system was tested by playing 250 rounds and analyzed with the use of the produced Q-table. The report finalizes with study limitations and recommendations for future work.

Keywords

Brain training; cognitive impairments; reinforcement-learning; the elderly; artificial intelligence.

1. INTRODUCTION

With the aging population, the number of middle-aged and elderly people with cognitive impairments is rapidly increasing (Kwon & So, 2014). As cognitive functions decline with age, these subjects become critical for elderly people. These impairments appear along a continuum from Mild Cognitive Impairment (MCI) or pre-dementia to Alzheimer's disease (Vogan, Alnajjar, Gochoo, & Khalid, 2020). Dementia gradually affects cognitive abilities such as memory, linguistic abilities, and thinking skills (Kwon & So, 2014). As the increase of an ageing population is unavoidable, the search for solutions for cognitive impairments is an ongoing topic in research.

Research has shown that the results of therapist-patient cognitive training and Human-Robot Interaction provide a reason for optimism (Vogan et al., 2020). Brain-training is suggested as an effective way to improve cognitive abilities to prevent dementia (Kwon & So, 2014; Villaverde et al., 2013). Besides this, users may be motivated by the option to independently improve one's memory and problem-solving skills can be a key motivator for users (Villaverde et al., 2013). By enabling users to train their brain individually, they might want to start with this in an early stage to take actions in possibly preventing dementia or similar cognitive decline.

Brain-training systems or applications offer a simple and effective alternative to expensive specialized assistive training software (Villaverde et al., 2013). Experts even mention that it is important that elderly users use brain fitness applications, as habits and dementia have a close relationship (Kwon & So, 2014).

It is suggested that brain-training programs dynamically should adapt to individual performance (Merzenich, 2007). To train the brain effectively, training must be at the "threshold" (i.e., the uppermost edge of ability) so that the brain can make gradual improvements. "*A software program built with algorithms that efficiently govern this adaptability is much more effectively dynamic than a human trainer or another delivery mechanism.*" (Merzenich, 2007).

Therefore, in this paper, we describe the development of a brain-training application that uses reinforcement learning to define the correct level of the user and to adapt different difficulty levels according to the performance of the user. The purpose of this project was to experience how reinforcement learning works and to build it into a self-developed concept which consisted of a memory game.

1.1 Related work

Research in educational psychology shows that instructions are most effective when it is in line with the level of the learner and meets their needs. As Vogan et al. (2020) describe, there is a level of instructional challenge that is precisely within a zone where learning is optimized. This zone is called the proximal zone of development. However, this level changes with time, instruction, and the emotional and physical state of the learner. Therefore, an individual receiving cognitive training should be continuously monitored (e.g., when there is a decay in cognitive function, instructions need to be altered).

Developments in artificial intelligence (AI) suggest that this is possible (Vogan et al., 2020). AI can be understood as an "intelligent agent" that perceives its environment, collects data, and uses this data to take appropriate actions to create maximum impact towards a particular goal (Bini, 2018).

Developments in AI, in combination with natural language processing, eye and face tracking, and gesture, and speech processing that can be embedded in robotic agents, will produce a steady increase of data (Luxton, 2019). This will optimize the capacity of cognitive training because the instructions will be given according to the proximal zone of development. Thus, rather than providing cold and standardized training, AI will never stop gathering information, closely aligning training to the needs of the "ever-changing individual" (Vogan et al., 2020).

In addition, many researchers have investigated the topic of AI within the use of games. For example, Nareyek (2004) and Cook et al. (2016) describe the use of AI in games in general. In these papers, the artificial agent is, however, described as only an element of the game that does not learn from past user interactions. For example, Nareyek describes the use of NPC's (Non-Playable Characters) in games and how such agents operate in games. Cook et al. give an overview of different roles agents have taken in well-known games, such as The Sims. However, these researchers did not describe learning agents at the core of the game, focused on optimizing or personalizing the player's experience. In their concluding remarks, Nareyek recommended focusing more on such player experience when employing machine learning for games.

In addition to AI-based game research, brain training games have been a well-investigated topic of research, especially Lumosity (Ballesteros et al., 2014; Navarro et al., 2013) and Dr. Kawashima's brain training games (Nacke, Nacke, & Lindley, 2009). Lumosity consists of different smaller games designed to train memory, attention, processing speed, and cognitive control. Dr. Kawashima's brain training games consist of similar minigames, aimed to train the cognitive abilities of its players. While these games have shown to be popular and effective to some extent, they were not designed for a population with cognitive impairments, such as Alzheimer's disease (Navarro et al., 2013). For this reason, Navarro et al. designed a brain training game that uses an intelligent agent to detect changes in the performance of users. This allowed the researchers to detect cognitive impairment at an earlier stage.

Another approach to employ machine learning for games is to adapt the difficulty level of the game towards the player (Imbeault, Bouchard, & Bouzouane, 2011). In their study, Imbeault et al. created a game for Alzheimer's disease patients that dynamically adapted its difficulty towards the level of the player. In their game, Alzheimer patients had to perform tasks related to everyday life, like making toast or coffee. The game made use of AI to analyze the in-game tasks performed by the user, as well as to adapt the difficulty of the tasks. They conclude with the note that such games, specifically designed for Alzheimer patients, should reflect everyday life, provide adequate feedback to the player and estimate the cognitive abilities of the player. For this project, we saw an opportunity to apply this knowledge into a practical design, as this is a small step towards a wider application of cognitive brain training games for elderly people with the use of AI.

2. METHODS AND MATERIALS

One way to keep Alzheimer patients engaged in daily activities, was found to be face memory training (Tak & Hong, 2014). Face recognition is one of the main human memory features and one of the symptoms of moderately severe Alzheimer's disease (Ahn, Santos, Wadhwa, & MacDonald, 2014; Bruce & Young, 1986; Tak & Hong, 2014). Similarly, face memory training was deemed effective for patients with brain injuries (Powell, Letson, Davidoff, Valentine, & Greenwood, 2008). For these reasons of effectiveness, we used face memory as the topic for the designed brain training system. In addition, we were inspired by the face memory game of Lumosity, as described by Ballesteros et al. (2014).

During our game, the elderly user is shown a series of illustrated faces on a screen. When this is finished, the user is shown a random face that appeared previously in the series. The user must then decide when the face appeared in the series and input their answer by pressing the corresponding number-key on their keyboard. The difficulty of this game, which is determined by the number of faces and the difference per face, is adjusted to fit within the user's zone of proximal development. This means that we aim to reach maximal effectiveness of the training by training the user at their uppermost edge of ability (Merzenich, 2007). Therefore, the game's difficulty must be adjusted for it to be not too easy and not too difficult for the user.

In our designed system, these adaptions are governed by a SARSA reinforcement learning model. Since our system will learn through interaction, instead of using a pre-made dataset, we chose to employ reinforcement-learning (RL). In RL, an agent takes actions in an environment (opposed to the user) (Argerich, 2020) by making observations and choosing actions accordingly. The AI learns which actions to take, given the current observations, by trying different actions in different contexts and inferring how good or bad that action was (relative to other actions taken) (MacGlashan, 2018). This is different from supervised learning for example, which uses fixed or pre-defined datasets which the system internalizes so that it knows what to do in future situations. Eventually, our system will learn which actions to take, through user interaction. This will be elaborated more in the next sections.

2.1 Learning algorithm

The designed brain training system made use of SARSA reinforcement learning. SARSA, an acronym for state-action-reward-state-action, is a type of reinforcement learning algorithm (Rummery & Niranjan, 1994). In reinforcement learning, the machine learning agent learns by finding a balance between exploration and exploitation (Kaelbling, Littman, & Moore, 1996).

The reinforcement learning environment consists of states, actions, and rewards. With SARSA reinforcement learning, the agent interacts with its environment by moving from state to state by performing actions. For some action results, rewards or punishments are given. The goal of a reinforcement learning agent is to maximize the total rewards (Rummery & Niranjan, 1994). Maximizing these rewards is done by Q-learning, which is the process of updating so-called Q-values, which represent the possible received reward in the next time step for taking a specific action in a specific state. The Q-values are updated with the following SARSA formula.

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

In this formula $Q(s_t, a_t)$ represents the current Q-value, corresponding with the current state and action. α represents the learning rate of the algorithm. This value determines how fast the algorithm learns, by giving weight to the current Q-value. r_{t+1} is the reward that is received after taking a certain action. γ represents the discount factor. This factor determines the importance of future rewards. When γ is close to 0, the system will only take current rewards into account, while when it is close to 1, it will account for future rewards (i.e. of the next state). γ can affect this by being multiplied with $Q(s_{t+1}, a_{t+1})$, the Q-value at the next state with the next action. Over time, the agent develops a policy to maximize the rewards with the use of this formula for Q-Learning and learns which actions are most rewarding (Sutton & Barto, 2018).

3. THE FACE MEMORY GAME

Eventually, the elderly user will be able to play the face memory game on a tablet as can be seen in Figure 1. The memory game shows several faces, and the user needs to remember at which position the current face was shown. For example, there are three faces shown (e.g., face 1 – face 2 – face 3), each with a duration of one second, then the system randomly picks one (e.g. face 2) of these shown faces and asks the user which face was shown (i.e. face 2). The system will compare the interaction score of the user to the pre-trained difficulty levels and defines a new level of difficulty accordingly.

In the game, features within the face can be changed such as the hair, face skin color, eyes, nose, and mouth. Furthermore, the number of faces shown can also be altered according to the difficulty level of the user. Each round, the game alters one of these difficulty types, based on the results and reward function. This way, the game has a difficulty level that fits with the user's abilities, and as the cognition would decline, the game will become easier.

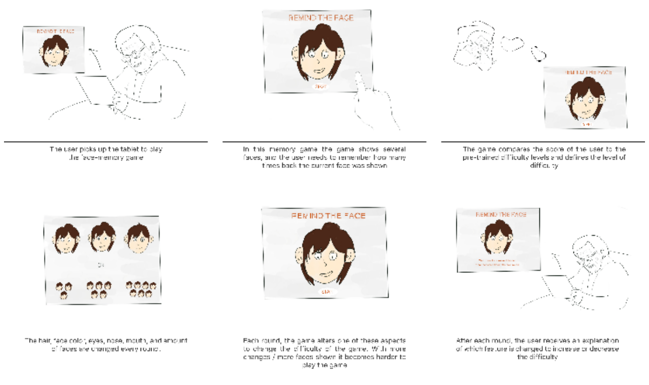

REMIND THE FACE
REMIND THE FACE
REMIND THE FACE
The user picks up the tablet to play the face-memory game
In this memory game the game shows several faces, and the user needs to remember how many times back the current face was shown
The game compares the score of the user to the pre-trained difficulty levels and defines the level of difficulty
REMIND THE FACE
REMIND THE FACE
The hair, face color, eyes, nose, mouth, and amount of faces are changed every round.
Each round, the game alters one of these aspects to change the difficulty of the game. With more changes / more faces shown it becomes harder to play the game
After each round, the user receives an explanation of which feature is changed to increase or decrease the difficulty

Figure 1 – User interaction scenario.

In the last step, we want to implement a form of Explainable AI (XAI) into our program. Explainable AI is desirable as advances in AI produce autonomous systems that will perceive, learn, decide, and act on their own. To increase understanding and trust, complex systems should explain their rationale and create an understanding of future behavior (Wallkotter, Tulli, Castellano, Paiva, & Chetouani, 2020). In our system, XAI is envisioned by providing an explanation of changed parameters and how the next round will be different compared to the former (e.g. the Q-values can provide information about the next steps, based on the current state and possible actions). This will be done in such a way that it is understandable for the elderly user. For example, the system could say "Unfortunately this answer was wrong, it seems that you have trouble with the number of faces shown, the next round less faces are shown, please try again!". This message could be based on the Q-values and reward functions in the system. However, to increase the user's autonomy, we might want to let the user choose what to change, based on system's suggestions. Then the user has the autonomy to change the difficulty level instead of the system doing it for them. This could be done by providing a text message that says "Well done! We think you are ready for a new challenge, there are two options: show me more faces, or have smaller changes in the faces. By clicking on one, you can start a new game!". This has not been implemented in the system yet.

3.1 Intelligent behavior and embodiment

The learning algorithm and the game itself were implemented as two separate programs, with one written in Python and another in Processing. Codes of both these programs can be found in Appendix I. These two programs interact with each other by writing values to text files, that are read by the other program. The game aspects of the system were programmed in Processing, while the machine learning aspects were written in Python. During a single round, after the system has presented the faces, the user inputs their answer by pressing a number-key on their keyboard (e.g. "2"). Processing then checks this answer and writes whether it was correct to a text file with either a '1' (i.e. "right") or a '0' (i.e. "wrong"). On another line in the same text file, the game writes how many rounds have passed since the start of the game. The game is then paused briefly, as it waits for input from the Machine Learning program. Every time a game round has passed, it uses the game's output, the correctness of the answer written to the text file, to determine an adjustment to the game's difficulty by using the ε-greedy and the updated Q-table. Then, the Python program writes the adjustment to another text-file, which is, in its turn, read by the game. Once the game recognizes that the learning algorithm has set a new difficulty, it adjusts the game's difficulty variables (i.e. the amount of change per face or the number of faces shown) to this input and a new round is started. How the system exactly determines a new difficulty level, is based on the state-action space and the SARSA algorithm of the system.

The proposed brain training system consists of a state-action space with eighteen different possible states and two actions. The following features were used.

- The number of faces shown (h: 3, 5, 7);
- The amount of change per face (f: 3, 4, 5);
- The answer of the user (a: right, wrong).

The two actions used for the system were the following.

- Changing the number of faces shown;
- Changing the amount of change per face.

The agent moves through the states, based on the answers of the user. When an answer is given, the learning agent determines the appropriate action. For example, when the user has provided a wrong answer, the value of h can be decreased from 5 to 3, decreasing the number of faces shown, lowering the difficulty. When the use provides a correct answer the next round, this value can be increased from 3 to 5 again, increasing the difficulty. When value of h is changed, the number of faces that are shown in series is adjusted. When the amount of change per face is changed, the amount of different face elements is changed. For instance, when the value of f is equal to 3, the skin color and eyes are the same for the first and second face, while the nose, mouth and hair will be different. The face after that, the third face, can have the same skin color and nose as the second face, while the mouth, hair, and eyes are different.

What difficulty type is changed (i.e., the number of faces, h, or the amount of change per face, f), is based on an ε-greedy policy (Rummery & Niranjan, 1994). This policy is represented by a value between 0 and 1, decreasing with the number of interactions. This policy allows the system to both explore and exploit the state-action space by either choosing an action based on the Q-Learning results or choosing it randomly. As the value of ε moves closer to 0 after every interaction, the actions will be chosen based on the Q-values in the Q-table more often. Together with the action choice, the reward is determined, and the Q-values are updated with the use of this reward. Based on the user's answer, these Q-values will increase when the user was right, changing the probability that this action will be chosen again in the future. The user is then presented with another round of the game with a changed difficulty level. An overview of the entire system can also be found in Appendix II.

3.2 Testing and analysis

To validate the functionality of the reinforcement-learning algorithm and its systemic implementation, the system should be trained to provide a baseline for a test case. Our system's state-action space was deemed small enough for training during use to be effective. For a sufficiently desirable user experience on first use however, we have prepared a non-zero Q-table (see Appendix III) that tells the reinforcement-learning algorithm how an average user reacts to the changes in the f and h values. To create this Q-table, we personally played the game for 250 episodes. As can be seen, this is an insufficient amount for a fully fitted table, but it does provide insights into user-system interactions and the more common states the user will encounter. For a full Q-table, more training would be needed but this was not realistic in the scope of this project.

An option for filling the Q-table efficiently would have been to write an algorithm based on our perceived difficulty of changes in the f and h values. We could quantify this perceived difficulty and use it to create an answering algorithm – a 'fake user'– to automate the training process and to enable going through the episodes in rapid succession, similar to the user models, as proposed by Tsiakas, Dagioglou, Karkaletsis, and Makedon (2016). Although the resulting Q-table would not be as accurate as a user-made one, it would be far less time consuming, and sufficiently accurate to improve the games' first use experience.

To further validate the functionality of the ML algorithm, the code was slightly modified. This allowed for the Q-values of the whole state-action-space to be printed to the terminal for each round, as well as the corresponding ε-greedy parameter and the given reward. This facilitated us with enough information to keep track of what the algorithm was doing, based on user input. We were able to conclude that the algorithm in its current state is fully functional.

For further validation of the written algorithm, we propose the further implementation of XAI. As explainability reveals the black-box functionality of the ML algorithm to the user, it can also serve to review the functioning of the system. By printing parts of the code during playing, the foundation for this was already created, but by developing this, we can further show the inner workings of the system.

4. CONCLUSION

The goal of this project was to experience how reinforcement learning works and to learn how to apply it in a self-developed concept. As this report and the memory game that we created show, we can say that we are satisfied with the result of this game given the time that was given in combination with our skills. We were able to build the game from scratch and implement reinforcement learning in an adequate and meaningful way.

As the Q-table in Appendix III shows, our implementation is not perfect. Several fields still show 0, and this means that for the less often played states – which are the harder levels – the system is not trained yet. As previously discussed, we have a clear idea on how our concept could be further improved, as well as how to implement explainability. Our design already highlights the importance of XAI. As we were validating the functionality of the system, the need for transparency of the black-box nature of ML algorithms became clear. Because we have programmed the RL functionality ourselves, we do not only grasp these inner working, but we also understand why designers express difficulty in prototyping with ML and envisioning new and purposeful uses for it (Dove, Halskov, Forlizzi, Zimmerman, 2017). We consider XAI as a possible solution for this.

We believe solutions like ours could contribute to countering cognitive decline. It offers the low-cost and easy to scale-up solution that computer systems are known for, while maintaining a level of personalization that would usually be lost in non-AI solutions. We believe ML will therefore play an increasingly important and meaningful role in the future of for example healthcare and education. Overall, this course offered a great inspiration and skillset for future projects and highlights the developments and relevance of ML, RL and XAI in this field.

5. DISCUSSION

Initially, this project started with the goal of creating an application that would suggest several brain-training games by using reinforcement learning. The different game suggestions would be based on different user performance levels. However, as this was one of our first encounters with programming the implementation of reinforcement learning, this was too ambitious. Therefore, the system was scaled down to one single game where a reinforcement learning system was built into.

However, there are still some limitations to the design. For example, we used a pre-existing SARSA learning algorithm instead of designing it ourselves and writing it from scratch. We chose the values of the learning rate, discount factor, and policy suggestion based on recommendations of our coach and did not have enough time to explore changes in these values. Since the system is trained through interaction, it was difficult to see immediate results as it takes time to train. For this course, we were able to get an understanding of applying a learning algorithm to a system. However, compared to our initial goals, we merely scratched the surface of reinforcement learning. For improvement of this concept, it would be wise to do further research on reinforcement learning algorithms and different applications to test which one is optimal.

Another limitation of the system could be regarding the visualization. Now, the game contains illustrated faces, which are not realistic. By using more realistic faces (for example, increase the quality of illustrations or using photographs of non-existing people where aspects change), the difficulty level might increase. Furthermore, this might improve the motivation of the user or enjoyment of the game. However, as these are assumptions it would be interesting to see how the results might change, and to evaluate the user experience. Furthermore, it is now possible that the system shows duplicate faces in the same series, due to time-limits we were not able to change this. For future work, the program should be altered so that each face is unique when they are shown to the user and no errors occur.

While testing the game, one observation was that when the difficulty was increased by changing the number of faces shown, the user might experience it differently from the assumption of the system. To elaborate, when the game shows 7 faces in a row and freezes on the last face shown, it is easier to remember compared to when the first or second face should be remembered. Currently, the system is not able to distinguish between these situations. One solution might be to divide the reward by the number of faces shown to align these two experiences. Due to lack of time, this was not implemented but would be suggested to enhance the user experience for future implementations.

In the current state of the game, the difficulty level of each round is increased whenever a user provides a correct answer. Within the SARSA algorithm, using rewards, the same reward is given for every difficulty level. This means that the same reward is given when a user reaches the highest difficulty level, as when they succeed in their very first round. To balance this and to reward the agent more appropriately for allowing the user to reach a higher level, rewards should be scaled to the difficulty levels. Future iterations of the system would then, for example, provide a higher reward when the user answers correctly when the number of faces shown is 5, compared to when the number of faces is 3. Provided the value of the amount of change per face remains the same in both rounds.

Looking at our initial vision: to help the elderly with brain-training activities to prevent cognitive impairment, some aspects could be improved. For example, to realize this game into an application. Here, multiple games should be suggested to keep the users interested. All these games should work according to a reinforcement learning system that optimize the difficulty levels according to each player. Also, XAI should be implemented to enable the user to understand why, and how games become easier/more difficult and to increase autonomy by providing explained choice possibilities. This means that the user receives information about what could become more difficult in the next round and is given the choice which aspect to change, based on this information.

As the game is trained by receiving data through interaction, we want to suggest using data of multiple users. Currently, the system was only trained by some team members to a certain point. However, it would be meaningful to have data from several different users to compare Q-values. Particularly user data of our target group, elderly, would be most meaningful for future development. This will make the suggestions more reliable, making it more effective for users. However, these assumptions should be validated with thorough testing and further development of the system.

Looking back at this project, we were able to come up with a method to implement reinforcement-learning into a concept within our vision. As this was our first experience building a system that incorporates reinforcement learning, we made progress in understanding how such a system makes decisions and how it learns. We experienced how to come from a concept to a structured and detailed plan to build it into processing and python. By building these into a concept, the information that was given in the lectures were made tangible. However, we do realize that we are far from experts in this field, but it enables us to come up with concepts that have a basic form of AI and to think along with more complex future projects as we have gained knowledge of this theory. Looking at the described theories, we tried to make a system accordingly. However, at this state we do not have sufficient data to say that elderly could be effectively helped with our system. In addition, future testing should show to what extent our proposed system limits cognitive decline and assists elderly users in daily brain training. It offers inspiration for future projects and highlights the relevance and developments in this field.

6. ACKNOWLEDGEMENTS

We want to thank our coach, dr. Kostas Tsiakas, for providing us with feedback during our design process and for providing us with his example code for reinforcement-learning functionality.

7. REFERENCES

Ahn, H. S., Santos, M. P. G., Wadhwa, C., & MacDonald, B. (2014). Development of Brain Training Games for a Healthcare Service Robot for Older People. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 8755, pp. 1–10). https://doi.org/10.1007/978-3-319-11973-1_1

Ballesteros, S., Prieto, A., Mayas, J., Toril, P., Pita, C., de León, L. P., Waterworth, J. (2014). Brain training with non-action video games enhances aspects of cognition in older adults: A randomized controlled trial. *Frontiers in Aging Neuroscience*, *6*(OCT), 1–14. https://doi.org/10.3389/fnagi.2014.00277

Bini, S. A. (2018). Artificial intelligence, machine learning, deep learning, and cognitive computing: what do these terms mean and how will they impact health care?. *The Journal of arthroplasty*, *33*(8), 2358-2361.

Bruce, V., & Young, A. (1986). Understanding face recognition. *British Journal of Psychology*, *77*(3), 305–327. https://doi.org/10.1111/j.2044-8295.1986.tb02199.x

Cook, M., Eladhari, M., Nealen, A., Treanor, M., Boxerman, E., Jaffe, A., … Swink, S. (2016). *PCG-Based Game Design Patterns*. Retrieved from http://arxiv.org/abs/1610.03138

Dove, G., Halskov, K., Forlizzi, J., & Zimmerman, J. (2017, May). UX design innovation: Challenges for working with machine learning as a design material. In Proceedings of the 2017 chi conference on human factors in computing systems (pp. 278-288).

Imbeault, F., Bouchard, B., & Bouzouane, A. (2011). Serious games in cognitive training for Alzheimer's patients. *2011 IEEE 1st International Conference on Serious Games and Applications for Health, SeGAH 2011*. https://doi.org/10.1109/SeGAH.2011.6165447

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, *4*(9), 237–285. https://doi.org/10.1613/jair.301

Kwon, S. M., & So, H. J. (2014). Investigating the needs of brain fitness programs in a mobile platform among older users: an initial user study. 한국 HCI 학회 학술대회, 98-103.

Luxton, D. D., & Riek, L. D. (2019). Artificial intelligence and robotics in rehabilitation.

Merzenich, M. (2007). Neuroscience via computer: brain exercise for older adults. *interactions*, *14*(4), 42-45.

Nacke, L. E., Nacke, A., & Lindley, C. A. (2009). Brain training for silver gamers: Effects of age and game form on effectiveness, efficiency, self-assessment, and gameplay experience. *Cyberpsychology and Behavior*, *12*(5), 493–499. https://doi.org/10.1089/cpb.2009.0013

Nareyek, A. (2004). AI in Computer Games. *Queue*, *1*(10), 58–65. https://doi.org/10.1145/971564.971593

Navarro, J., Zamudio, V., Doctor, F., Lino, C., Baltazar, R., Martinez, C., … Gutierrez, B. (2013). Game based monitoring and cognitive therapy for elderly. *Workshop Proceedings of the 9th International Conference on Intelligent Environments*, *17*, 116. https://doi.org/10.3233/978-1-61499-286-8-116

Powell, J., Letson, S., Davidoff, J., Valentine, T., & Greenwood, R. (2008). Enhancement of face recognition learning in patients with brain injury using three cognitive training procedures. *Neuropsychological Rehabilitation*, *18*(2), 182–203. https://doi.org/10.1080/09602010701419485

Rummery, G. A., & Niranjan, M. (1994). *On-line Q-Learning Using Connectionist Systems*. Cambridge, UK: University of Cambridge, Department of Engineering.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd ed.). Retrieved from https://lccn.loc.gov/2018023826

Tak, S. H., & Hong, S. H. (2014). Face-name memory in Alzheimer's disease. *Geriatric Nursing*, *35*(4), 290–294. https://doi.org/10.1016/j.gerinurse.2014.03.004

Tsiakas, K., Dagioglou, M., Karkaletsis, V., & Makedon, F. (2016, November). Adaptive robot assisted therapy using interactive reinforcement learning. In International Conference on Social Robotics (pp. 11-21). Springer, Cham.

Villaverde, L. M., Smith, S. R., & Kurniawan, S. (2013, October). Brain-training software for stroke survivors. In Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility (pp. 1-2).

Vogan, A. A., Alnajjar, F., Gochoo, M., & Khalid, S. (2020). Robots, AI, and cognitive training in an era of mass age-related cognitive decline: a systematic review. IEEE Access, 8, 18284-18304.

Wallkotter, S., Tulli, S., Castellano, G., Paiva, A., & Chetouani, M. (2020). *Explainable Agents Through Social Cues: A Review*. 1–26. Retrieved from http://arxiv.org/abs/2003.05251

APPENDIX

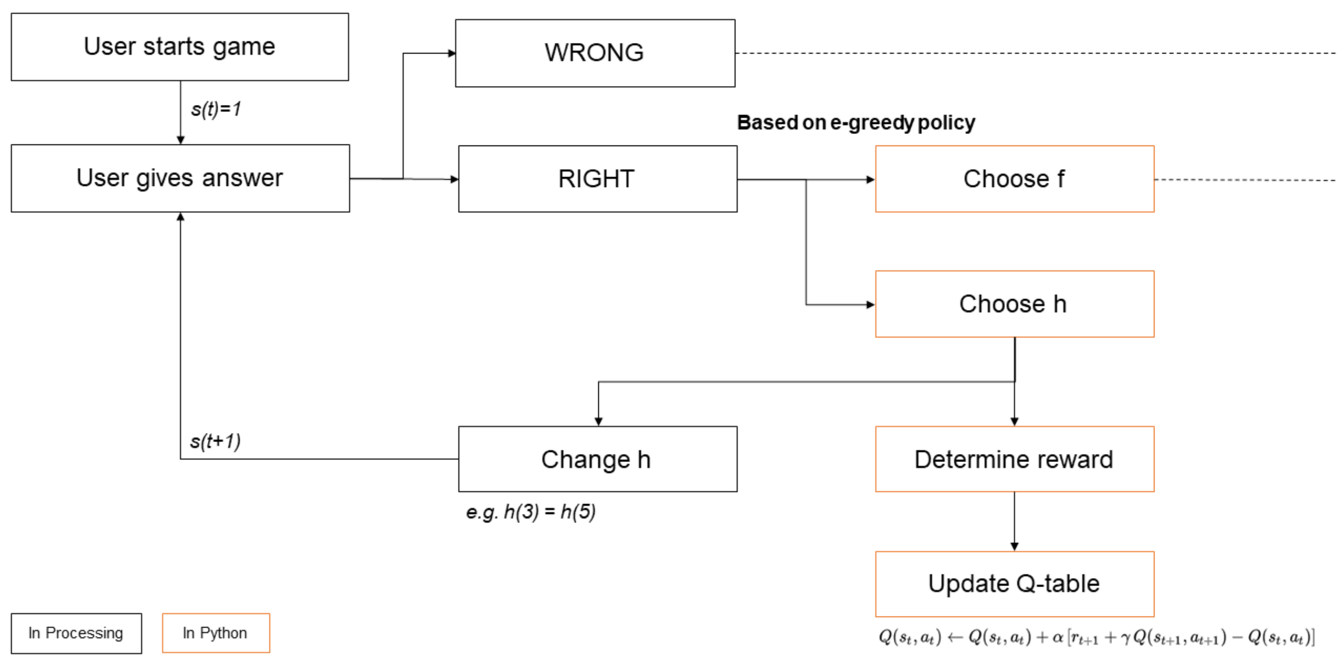
I. Code

The code for our Processing game and Python 3 script can be found on GitHub. The page also includes instructions on how to run the code, and can be found here: https://github.com/jorritvanderheide/eib_reinforcement_learning.

The files will also be included with this submission on Canvas.

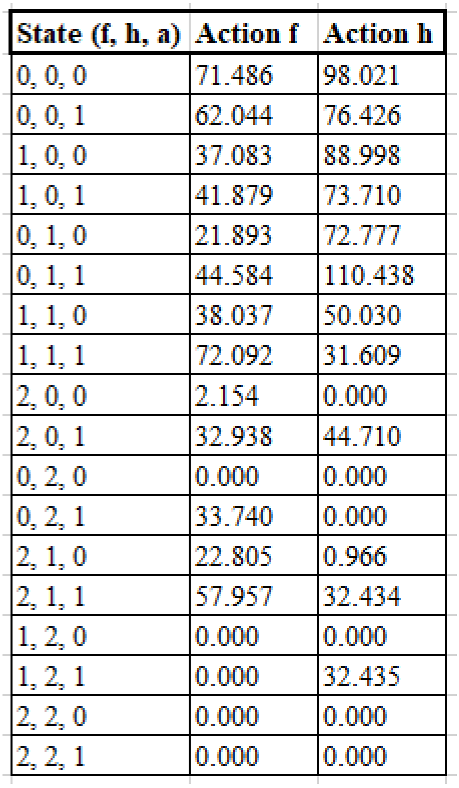
II. System diagram

The following diagram is an overview of how our system functions.



III. Q-table

This is the Q-table after 250 episodes.



State (f, h, a)	Action f	Action h
0, 0, 0	71.486	98.021
0, 0, 1	62.044	76.426
1, 0, 0	37.083	88.998
1, 0, 1	41.879	73.710
0, 1, 0	21.893	72.777
0, 1, 1	44.584	110.438
1, 1, 0	38.037	50.030
1, 1, 1	72.092	31.609
2, 0, 0	2.154	0.000
2, 0, 1	32.938	44.710
0, 2, 0	0.000	0.000
0, 2, 1	33.740	0.000
2, 1, 0	22.805	0.966
2, 1, 1	57.957	32.434
1, 2, 0	0.000	0.000
1, 2, 1	0.000	32.435
2, 2, 0	0.000	0.000
2, 2, 1	0.000	0.000