TÉCNICO LISBOA

Player Preference Extraction From In-Game Behavior

André Pais Borges de Macedo Leite

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor: Prof. Carlos António Roque Martinho

Examination Committee

Chairperson: Prof. David Martins de Matos
Supervisor: Prof. Carlos António Roque Martinho
Member of the Committee: Prof. Sandra Pereira Gama

November 2021

Acknowledgments

I would like to thank my dissertation supervisor, Prof. Carlos Martinho, for his insight, support, guidance, and sharing of knowledge that has made this work possible. Thank you to my entire family and my friends for their support during these difficult times. I would also like to give a special mention to Victor Ribeiro for his patience during our long discussions and finally a special thanks to my friends at Los Cotons for being there with me every day.

Abstract

Procedural Content Generation is able to generate content tailored to players, but we need to discover the player's preferences to achieve that. This work aims to tackle how to extract and create a machine learning model of the player's preferences from gameplay collected data. To achieve this, a single-player offline game was created, where we placed carefully crafted challenges, based on six of the seven BrainHex classes studied, from which we removed Socializer as it does not fit our type of game, ensuring we matched the players preferences we were trying to measure.

The player's gameplay data was extracted from the their interaction with challenges and the environment, and parsed to fit our machine learning needs. The parsed data was then used with a variety of machine-learning algorithms, like Naive Bayes, Decision Trees, and K-Means to predict future players' gaming preferences.

The dataset was replicated six times, one for each BrainHex class, and separately used to train different machine learning models. Even with a very limited sample size of 30, (n=24 for the training set and n=6 for the validation set), our models reported a high accuracy in identifying the BrainHex classes of the players for five of the six datsets. The highest accuracy for each dataset in validation was: 100% for Conqueror, 100% for Achiever, 100% for Mastermind, 83.33% for Survivor, and 66.66% for Daredevil.

Keywords

Player Models; Personality; Machine learning; Data Mining.

Resumo

A Geração de Conteúdo Procedimental é capaz de gerar conteúdo adaptado aos jogadores, mas precisamos de descobrir as preferências do jogador para o conseguir. Este trabalho visa abordar a forma de extrair e criar um modelo machine learning das preferências do jogador a partir dos dados de jogo recolhidos. Para o conseguir, foi criado um jogo offline e singleplayer, onde colocámos desafios cuidadosamente elaborados, com base em seis das sete classes BrainHex estudadas, das quais retirámos o Socializer por não se adequar ao nosso tipo de jogo, a correspondência com as preferências dos jogadores que estávamos a tentar medir

Os dados de jogabilidade do jogador foram extraídos da sua interacção com os desafios e o ambiente, e adaptados às nossas necessidades de machine learning. Os dados foram então utilizados com uma variedade de algoritmos de machine learning, como Naive Bayes, Decision Trees, e K-Means, para prever as preferências de jogo dos futuros jogadores.

O conjunto de dados foi replicado seis vezes, um para cada classe BrainHex, e utilizado separadamente para treinar diferentes modelos de machine learning. Mesmo com um tamanho de amostra muito limitado de 30, (n=24 para o conjunto de treino e n=6 para o conjunto de validação), os nossos modelos relataram uma alta precisão na identificação das classes BrainHex dos jogadores para cinco dos seis conjuntos de dados. A maior precisão para cada conjunto de dados em validação foi: 100% para Conqueror, 100% para Achiever, 100% para Mastermind, 83,33% para Survivor, e 66,66% para Daredevil.

Palavras Chave

Modelos de Jogador; Personalidade; Aprendizagem; Data Mining.

Contents

List of Figures

List of Algorithms

Acronyms

CSV	Comma-Separated Values
DGD	Demographic Game Design
FFM	Five Factor Model
GMP	Quantic Foundry's Gamer Motivation Profile
GEQ	Game Experience Questionnaire
ID3	Iterative Dichotomiser 3
IST	Instituto Superior Técnico
KNN	K-Nearest Neighbors
LWC	Legend of the Warrior's Crystals
MBTI	Myers & Briggs' Type Indicator
NPC	Non Playable Character
PCG	Procedural Content Generation
UID	Unique Identifier
WEKA	Waikato Environment for Knowledge Analysis

1

Introduction

Contents

1.1 Motivation

With the games industry's constant growth [1], developers are always searching for new ways to make their games stand out in such a crowded environment and appeal to the highest number of potential players.

Choosing what type of game to make can become a problem if the genre ends up being unpopular, giving less prospects of turning a profit. This reduces the likelihood of certain types of game being made, since a player may see the game's genre and automatically assume they won't like it. To solve this problem, we can try and capture the widest range of preferences possible, by making a game that adapts to each player, considerably increasing the pool of player types the game targets. There have been various solutions presented by developers throughout history, with most of them relying on psychological theories to adapt to the player. Using personality models to adapt a game to the player has been shown as a working solution to the problem [1], but multiple people and organizations have tried to find a better way of expressing player's preferences, this being how player type models were born. The development of player type models sparked some research on how to adapt a game to a given player's preferences; however, on the other hand, not much research has been made on how to figure out the player's preferences inside the game environment. This is a crucial step because asking an individual to answer a questionnaire before letting them start playing a game can be not only very intrusive and considered a hassle, but it could also make them give up on playing the game. Another point to consider is that the research which tackles this specific problem is focused on simpler player satisfaction models, which can be too general to use in adapting to a given player's preferences.

Our work will then focus on tackling this idea to extrapolate players' playstyle preferences from data gathered in-game.

1.2 Problem

The problem we are trying to tackle is how to extract the players' preferences to generate tailored content. In the past, some games have resorted to straight out questioning the player about their likings [2] [3], which is a very invasive approach, while others tried to understand how the players behave by their actions or inactions in-game by analyzing game telemetry data. It's this second approach which we chose to tackle.

Generating and analyzing this data has become a crucial topic in this field of study [2,3], with multiple approaches, spanning several different personalities' and player types' models being studied and tested

[1] https://www.statista.com/statistics/292056/video-game-market-value-worldwide/
[2] Silent Hill: Shattered Memories https://www.konami.com/games/eu/en/products/shsm/
[3] Until Dawn https://www.supermassivegames.com/games/until-dawn

[1, 4–6]. Most successful approaches rely on retrieving data throughout gameplay, analyzing it later with different machine learning algorithms to guess which preferences the player reveals. This can then be used to generate new content tailored to that specific player procedurally.

1.3 Hypothesis

In this work, we propose a methodology for collecting and processing game data to extrapolate the player's preferences for later use in adaptive content. It will explore the hypothesis that we can derive the player's preferences by gathering data from meaningful game data-points, like the player's options and decisions and how they execute them.

As such, we propose a method to extract the player's preferences from their in-game behavior. To achieve this, we will create challenges tailored to different types of player preferences, collect the data about how the given players interact with the game environment, and finally apply different data-mining techniques to obtain a model capable of deciphering the player's preferences from their gameplay.

1.4 Contributions

This work's main contributions consist of research on collecting and processing gameplay data to analyze and generate a predictive model of player preferences. With existing methods working mainly around answering questionnaires, we will try to simplify and enhance the experience from the player's side, by having them play a game to obtain their preferences. We will also do a literature review on personality and player models, procedural content generation, and data-mining techniques.

To achieve this goal, we will create a game which will be designed with the intent of allowing the player to express their gameplay preferences through their actions. To measure this, we will collect information about the actions the player took inside the game, with the intent of creating a dataset which matches to the player's real preferences obtained through a questionnaire. Using the dataset, we will employ several machine learning algorithms to create models which will be able to predict future players' gaming preferences. This models will be validated using additional data.

1.5 Document Outline

This document will be structured as follows:

Chapter 1 will introduce the motivation, problem, and hypothesis of our work.

Chapter 2 will consist of a literature review on the different topics tackled throughout the document, such as personality and player models, data-mining techniques, and past works related to our study.

Chapter 3 will describe our solution consisting of the architecture and methodology.
chapter 4 will describe the results obtained from our experiment.

2

Related Work

Contents

This section will tackle different subjects that will help us in guiding our work. We will tackle three different topics, Player Profiling, Procedural Content Generation (PCG), and Data Mining. In Player Profiling, we will examine different methods of categorizing players and how we can use such models to determine player behavior. In the PCG topic, we will discuss how PCG can be divided into different categories and how it will relate to our work. Lastly, in Data Mining, we will discuss different data-mining algorithms, which are more relevant to our work, and how they can help us categorize players.

2.1 Personality Theories

Personality theories/models are taxonomies that try to classify people by the way they interact with and act upon the world. With every person being unique in some way or another, these models categorize people's behavior using one or more categories. This section will explain such theories, such as Myers & Briggs' Type Indicator (MBTI) and the Five Factor Model (FFM). We decided to focus on these two theories since there have been many studies involving games with them, and many player models have been created based on them.

2.1.1 Myers & Briggs' Type Indicator

MBTI [7] is an introspective, self-report questionnaire based on Carl Jung's theory and developed by Katharine Cook Briggs and her daughter Isabel Briggs Myers. It classifies individuals according to four psychological preferences relating to how they perceive the world around them: extraversion-introversion, sensing-intuition, thinking-feeling, and judging-perceiving [7].

Extraversion – Introversion: signifies the source and direction of a person's expression. An extravert has it mainly in the external world, while an introvert has it mainly in their inner world.

Sensing – Intuition: represents the method by which someone acquires information. Sensing people work better with information in details and facts, while those who prefer intuition favor information less reliant upon the senses, building abstract theoretical models, patterns, and connections.

Thinking – Feeling: represents how a person processes information. Thinking tends to relate to logical reasoning, while feeling is usually related to decisions based on emotion.

Judging – Perceiving: reflects how a person acts in the outside world. Judging categorizes a person who organizes all of their life events and sticks to their plans while perceiving individuals are more inclined to improvise and explore alternative options.

2.1.2 Five-Factor Model

The FFM, also known as the Big Five personality traits, is a proposed categorization of individuals' personalities. It was formed by applying factor analysis to several independent sets of surveys on personality data [8]. This analysis revealed similarities between several different verbal descriptions of personality traits, combining them into five main factors, openness to experience, conscientiousness, extraversion, agreeableness, and neuroticism [8].

Openness to experience: Individuals classified as having high openness show an appreciation for art, emotion, adventure, unusual ideas, imagination, curiosity, and variety of experiences, and usually search for intense and euphoric experiences. On the other hand, those categorized with a low score in openness are labeled as data-driven and pragmatic.

Conscientiousness: Individuals classified as having high conscientiousness are more likely to be labeled as self-disciplined, dutiful, and focused. On the other hand, those with low conscientiousness tend to be labeled as spontaneous, sloppy, and unreliable.

Extraversion: Individuals classified as having high extraversion are driven by external stimuli, prefer activities related to the outside world, and enjoy interacting with other people. On the other hand, those with a low score in this category (introverts) are less socially involved and usually labeled as quiet and shy.

Agreeableness: Individuals classified as having high agreeableness have a tendency to be labeled as considerate, generous, trustworthy, and are more willing to sacrifice their happiness for others. On the other hand, those who score low in this category tend to be more self-centered, placing their interests above all else.

Neuroticism: Individuals classified as having high neuroticism show a higher probability of experiencing negative emotions like hatred, anxiety, and depression. On the other hand, those who score low in this category are generally calmer, emotionally stable, and mostly clear of persistent negative feelings.

2.2 Player Models

Player's preferences are reflected by their actions in-game and by the choices of games and content they engage with. These traits are what player type models try to categorize and explain. Some of these player models are based upon the personality models discussed above (BrainHex and Quantic Foundry's Gamer Motivation Profile (GMP)), while others are based on direct observation of player behavior (Bartle taxonomy of player types), and lastly the Marczewski's Player and User Types Hexad which is based on research about human motivation as well as another player model (Bartle taxonomy of player types).

2.2.1 Bartle taxonomy of player types

Richard Bartle developed Bartle Player Types [9], one of the first models to classify players according to their preferred gameplay actions. A study conducted to analyze the players' behavior in a MUD (Multi-user Dungeon) created four categories: Achievers, Killers, Explorers, and Socializers. Two main axes of interest, Players-World, and Acting- Interacting, derive the four different categories [9].

In Figure 2.1 we can observe the relationship between each player type (in each quadrant) and the corresponding dimensions in the graph's axes.

Achievers: act upon the world and desire anything representing a concrete measurement of success in the game, such as levels and equipment.

Killers: act upon players and enjoy competition, particularly against real players rather than with NPCs (Non-Player Character).

Explorers: interact with the world, prefer immersion, and to discover areas in the game world. They are annoyed by time restrictions as they like to move at their own pace. Additionally, they like fiding glitches and easter eggs.

Socializers: interact with players, preferring the social aspect of the game, interacting with other human players, and sometimes NPCs, more than the game itself.

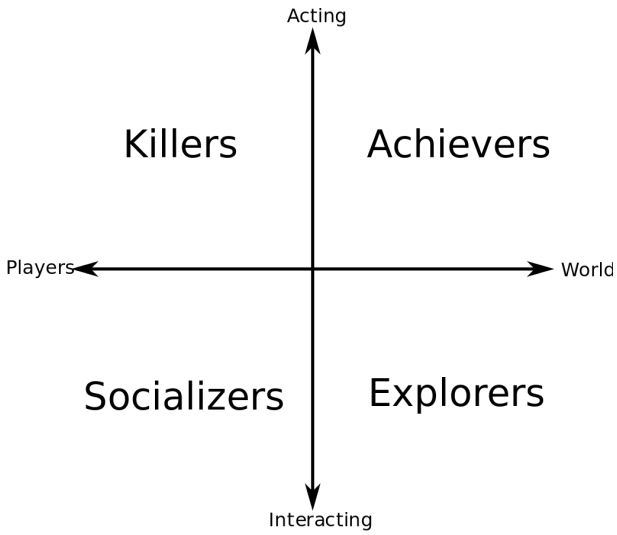


Figure 2.1: Bartle Types

2.2.2 Quantic Foundry's Gamer Motivation Profile

GMP [10], initially developed in 2015 by Nick Yee and Nicolas Ducheneaut, categorizes players using twelve motivations inspired by other works like the FFM, grouped in pairs by factor analysis [10].

Immersion: Players that exhibit high scores in this motivation tend to like games that let them engage in compelling narratives, settings, and customization options. Contrarily, those that manifest low scores favor gameplay mechanics and are less reliant on gameplay experiences to enjoy a game. The two low-level motivations are:

Fantasy: The desire to become someone else, somewhere else.

Story: The importance of an elaborate storyline and interesting characters.

Creativity: Players who score high in this category take joy in experimenting with the game world and transforming it with their designs and customizations. Contrarily, those with low scores are more practical in their gaming style and experience their game worlds without trying to change them. The two low-level motivations are:

Design: The appeal of expression and deep customization.

Discovery: The desire to explore, tinker, and experiment with the game world.

Action: Players that exhibit high scores in this motivation are generally more aggressive and enjoy dropping into action with dramatic visuals and effects. Contrarily, those with low scores are more relaxed and prefer slower-paced games. The two low-level motivations are:

Destruction: The enjoyment of chaos, mayhem, guns, and explosives.

Excitement: The enjoyment of games that are fast-paced, intense, and provide an adrenaline rush.

Social: Players who score high in this category like to interact with other players, preferring games that let them engage in social activities. Contrarily, those with low scores prefer games where interacting with other players is not necessary and value independence. The two low-level motivations are:

Competition: The enjoyment of competition with other players (duels or matches).

Community: The enjoyment of interacting and collaborating with other players.

Mastery: Players who exhibit high scores in this motivation cluster prefer complex gaming experiences to plan actions with strategic depth. Contrarily, those with low scores like being spontaneous with their decisions, opting for games that allow them to make mistakes. The two low-level motivations are:

Challenge: The preference for games of skill and enjoyment of overcoming difficult challenges.

Strategy: The enjoyment of games that require careful decision-making and strategic thinking.

Achievement: Players who score high in this category enjoy collecting items, achievements, and any other kind of collectible, regardless of the time needed to acquire them. Contrarily, those with low scores are not bothered with collecting things. The two low-level motivations are:

Completion: The desire to complete every mission, get every collectible, and discover hidden things.

Power: The importance of becoming powerful within the context of the game world.

Three higher-level clusters are formed by grouping the six motivation clusters in pairs [10].

Immersion-Exploration: "covers different ways of relating to the story and design of the game world, whether via the narrative, the characters, or exploring and customizing the game world."

Achievement-Mastery: "covers different ways of progressing through and attaining power within the construct of the game world, whether this is leveling up, completing all its missions, or gaining mastery through practice."

Action-Social: "covers more energetic and gregarious modes of gameplay, seeking out arousing gaming experiences whether this is from playing with other people, intense gameplay, or dramatic destruction."

The authors later published in a blog post that they found a correlation between these three clusters and the FFM with Immersion-Exploration corresponding to Openness, Achievement-Mastery with Conscientiousness, and Action-Social with Extraversion [10].

2.2.3 Marczewski's Player and User Types Hexad

Marczewski's Player and User Types Hexad is a player and user type model directed to gamification systems. The model talks about six different types of users, Achiever, Socializer, Philanthropist, Free Spirit, Player, and Disruptor, with two of them, Player and Disruptor, labeled as not having very concrete motivations [11]. Each user type is related to one motivation Figure 2.2:

Achiever: Motivated by Mastery, always looking to improve and overcome challenges.

Socializer: Motivated by Relatedness, they are looking to interact with other users and create social situations.

Philanthropist: Motivated by Purpose and Meaning, they are characterized by being altruistic.

Free Spirit: Motivated by Autonomy and self-expression, they are looking to create and explore.

Player: Motivated by Rewards, they will do what is necessary to collect rewards.

Disruptors: Motivated by Change, they are looking to disrupt the system directly or through other users to achieve some change, be it positive or negative.

The model also presents us with another more detailed view of its user types by subdividing the Player and Disruptor types into four subtypes, each closely linked to one of the four main user types [11].

2.2.4 BrainHex

BrainHex is a satisfactory player model created by *International Hobo Ltd*, based on studying neurobiological research papers and directly influenced by the Demographic Game Design (DGD)1 survey results (which resulted in the DGD1 model) and the DGD2 survey [12]. The BrainHex model [13] uses

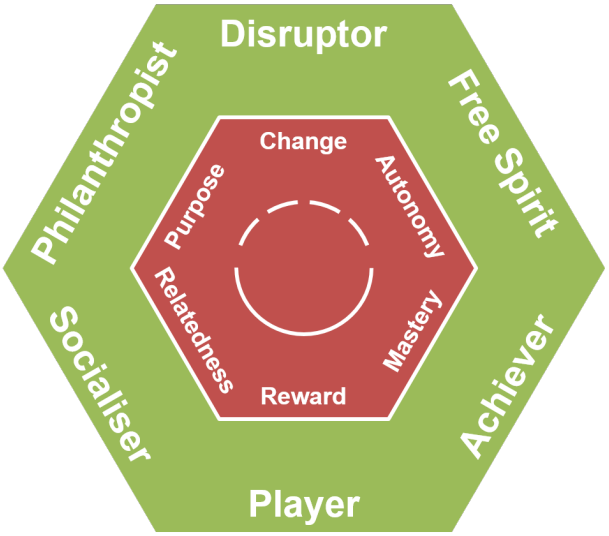


Figure 2.2: Player and User Types

seven archetypes, where each one links to a key element in the human nervous system, to define players' motivation and behavior in-game Figure 2.3. The model also defines exceptions as opposites of each class. These refer to what the player dislikes the most [13].

Seeker: Likes exploring the game world to find strange and wonderful or familiar things, showing curiosity, sustained interest, and love to stimulate their senses. The exception: **No Wonder**, dislikes searching for things, prefers clearly defined tasks.

Survivor: Likes escaping from hideous and scary threats while staying at the edge of fear to then feel safe again. The exception: **No Fear**, does not enjoy feeling afraid, prefers a safe environment, and feeling in control.

Daredevil: Likes the thrill, excitement, and risk-taking involved with things like moving at high speed while still in control and negotiating dizzying platforms. The exception: **No Pressure**, dislikes performing under pressure, prefers to take their own time to make the right decision.

Mastermind: Likes to solve puzzles and devise strategies while making the most efficient decision. The exception: **No Problems**, dislikes solving puzzles or finding solutions without explicit instructions.

Conqueror: Likes to struggle until achieving victory over difficult opponents and beating other players. They channel their anger to overcome difficulties and show superiority. The exception: **No Punishment**, dislikes struggling to overcome seemingly impossible challenges and repeating the same task repeatedly.

Socializer: Likes to socialize and help people they trust. They are usually trusty and get angry at

those that abuse their trust. The exception: **No Mercy**, does not care about hurting other players' feelings or prefers to keep their own company and does not enjoy playing with other people.

Achiever: Likes to complete and collect everything inside the game and become obsessive at overcoming a very distant goal. The exception: **No Commitment**, dislikes being asked to complete everything, preferring to pick and choose which tasks to attempt, or simply messing around with the game.

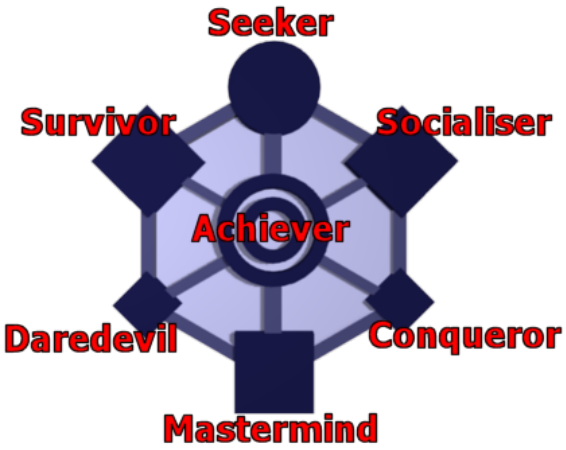


Figure 2.3: BrainHex Classes [13]

Upon taking the survey, each player will be assigned a score, from a scale of -10 to 20 to each class in accordance with their expressed preferences. This means that a given player is not only defined by their main class (highest score on the survey), subclass (second-highest score on the survey), and/or exception (negative score in a given category) but by the score they obtained in each category. A given individual's BrainHex score can look like:

$$Mastermind : 17|Socialiser : 16|Daredevil : 10|Achiever : 10|Conqueror : 10|Seeker : 6|Survivor : -4$$

This score means the player has a main class of Mastermind and subclass of Socializer.

2.3 Data Mining

Data mining is the application of algorithms with the purpose of extracting information (discovering patterns and knowledge) from data [14]. In games, telemetry refers to collecting player data stored for or by the developers. Game designers can then analyze telemetry data to understand player behavior and adapt the game content to the player's preferences or needs. Developers can use several data mining algorithms to analyze player behavior; each has its advantages and disadvantages and might

require/perform better using different data types. This section will explore how different data mining algorithms work and how we will evaluate them to choose the one that generates the best model, as previously mentioned in the hypothesis. The selection of the algorithms presented here reflects their relevance to the study conducted in this paper.

2.3.1 Algorithms

2.3.1.A Decision Trees

Decision tree is a machine learning algorithm that creates a tree-like structure with nodes and leaves. Each node represents a conditional control statement where a given attribute is tested, usually comparing its value against constant. Each leaf represents the classification of one instance, a set of them, or a probabilistic distribution. Finally, branches signify the connection between different nodes or nodes and leaves of the tree [14].

The process of classifying an unknown object starts at the root of the tree, where a test occurs based on the parameters of the given object, directing it down through the appropriate branch based on the result of the node's test. When the object finally reaches a leaf, it is classified as per the leaf's assigned class.

The Iterative Dichotomiser 3 (ID3) algorithm, developed by Ross Quinlan [15], starts building the tree from the root node (top-down) and creates a new node based on the best present feature, selecting it based on information gain.

C4.5

C4.5 is a decision tree algorithm developed by Ross Quinlan [16], extending the already existent ID3 algorithm. Much like the ID3 algorithm, C4.5 also generates decision trees based on the information gain theory.

The C4.5 algorithm starts at the root node by selecting the attribute with the highest information gain to split the node. After doing the split, we recursively look into the next attribute with maximum information gain and generate the node for the given branch with it.

The algorithm has a few base cases that it always checks for:

1: When all the samples in the given node's possibility list belong to the same class, a leaf node is generated instead of a decision node.

2: The features in the possibility list do not provide any information gain. It proceeds to create a decision node higher up in the tree.

3: A new instance of a class is encountered. It proceeds to create a decision node higher up in the tree.

Random Forests

Algorithm 2.1: C4.5 Algorithm [17]

1. Check for base cases
2. For each attribute a
3. Find the normalized information gain from splitting a.
4. Let a_best be the attribute with the highest information gain.
5. Create a decision $node$ that splits on a_best.
6. Recur on the sublists obtained by splitting a_best, and add those nodes as children of the $node$.

Random Forests are an ensemble learning method for classification, regression, and other problems that was first developed by Tin Kam Ho [18]. It works by creating a large number of decision trees. For classification problems, the Random Forest output is the class chosen by the majority of trees, correcting for the overfitting of decision trees.

In 2010, a study [19] was conducted to predict the last level a player would play before stopping playing the game or, if they finish the game, how long they will take to finish it. The study covered the use of the machine learning tool Waikato Environment for Knowledge Analysis (WEKA) and tested several machine learning algorithms, including C4.5 decision trees.

2.3.1.B K-Means

A cluster is a set of data objects where each object is similar to others in the same cluster and dissimilar to objects in different clusters. Clustering is an unsupervised classification algorithm by which we arrange the objects in clusters fig. 2.4.

In the K-Means algorithm, we start with a training set composed of N samples, and our goal is, given a value of K, partition the data into a K number of clusters. We end up with K centroids (cluster centers) where each point in a given cluster is always closer to its cluster centroid than to another cluster's centroid. Each centroid is represented by the mean value of all points contained in the given cluster [14].

The standard K-Means algorithm starts by initializing the centroids with random values and assigning each point to the closest centroid. It then computes the new centroid based on the mean value of all data points belonging to the cluster. It then repeats from step one (but now with the new centroids) until convergence. Since convergence is not guaranteed with this algorithm, we can have a stop condition to prevent an infinite run. With this approach, each run of K-Means can results in different clusters since the centroids initialization is random.

An example of using k-Means to cluster user behavior from in-game data can be seen in [20]. In the paper, published in 2012, the authors used k-Means to extract user behavioral patterns from high dimensional data collected from two major commercial game titles.

[1] https://wikipedia.org/wiki/K-means_clustering#/media/File:ClusterAnalysis_Mouse.svg

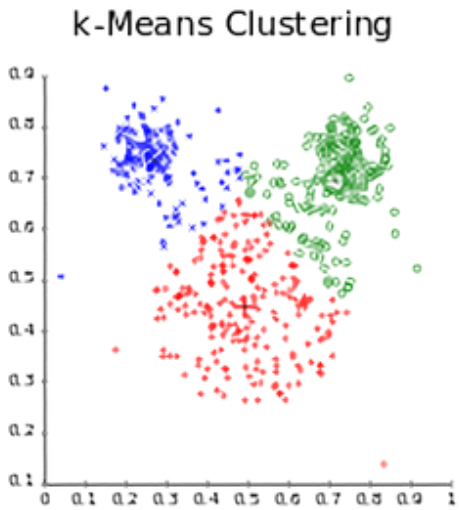


Figure 2.4: K-means Clusters [1]

2.3.1.C Naive Bayes

Naive Bayes classifier is based on Bayes' theorem and requiring substantial attribute independence, is one of the most practical learning methods available. It works by taking as an input a feature vector X to predict the corresponding class Y. This means that given a data point $X = (x1, x2, ..., xn)$, we want to figure out the odds of the class Y being y [14].

$$P(Y = y|X = (x_1, x_2, x_n))$$

However, we can only obtain $P(X|Y)$, $P(Y)$, and $P(X)$, so we resort to Bayes Theorem to get $P(Y|X)$.

$$P(Y|X) = \frac{P(X|Y)P(y)}{P(X)}$$

There is, however, a problem that arises with this approach, since with a growing number of features, the number of parameters grows exponentially. To tackle this problem, we assume that all features are independent, allowing us to classify a given instance with much fewer parameters.

$$P(Y|X) = \frac{P(Y)\prod_i P(X_i|Y)}{P(X)}$$

With this approach we reduce the number of parameters needed from $2^{k+1} - 1$ to $2k$.

With the use of Naive Bayes, we will need to discretize any continuous variables that we might collect. There is also the option of using a known distribution to fit the data into. However, this method depends on the data we collect and will be explored later if the need arises.

2.3.2 K-fold Cross-Validation

K-fold Cross-Validation is a model validation technique to test how a given learning algorithm will generalize to a detached dataset. This technique consists of splitting the data into equally sized k partitions, reserving one of the given partitions for validation, and training the machine learning algorithm with the rest of the k-1 partitions. With the split made, we run the machine learning algorithm k times, one for each time a different partition is selected for validation. This means that if we use 10-fold cross-validation on K-Nearest Neighbors (KNN), we will randomly split the data into ten partitions, select one partition k_i and run KNN over all k_j partitions where $j \neq i$. We then repeat this until we have selected all ten partitions for i. After finishing, we can average the results and obtain a good idea of how KNN will perform on an independent data set [14] fig. 2.5. The method described above uses the leave-one-out strategy, and by changing the number k of partitions we use, we can change the split percentage.

Figure 2.5: K-fold Diagram [2]

2.3.3 WEKA

The WEKA [3] [14] is a machine learning tool developed at the University of Waikato in New Zealand, including a big library of machine learning algorithms and a way to visualize the input data and the results obtained applying the learning algorithms. WEKA provides us a quick and easy way of interpreting our data and obtaining performance metrics of our classifiers. It only has one prerequisite: feeding it the data in one of its supported formats.

[2] https://www.researchgate.net/figure/Diagram-of-k-fold-cross-validation-with-k-10-Image-from-Karl-Rosaen-Log_fig1_332370436

[3] https://www.cs.waikato.ac.nz/ml/weka/

We will also use one of WEKA's features, which allows us to compare different learning algorithms' performance applied to our data.

2.4 Previous works

As we mentioned before, not much work has been made regarding the identification of player preferences through in-game behavior. In this section we will present some past studies on which we based different parts of our work.

2.4.1 Keirsey Temperament Model

The work done by Capelo [21] explored the idea that player actions and choices could be an indication of personality. They based their work on the Keirsey Temperament Model, developing a methodology to design scenarios that allowed the collection of game data from which an inferring system to classify players was created. His work tackled the problem of player choice to inform about their preferences, by making a gamemode inside the game Minecraft, where players could experience two different rooms each based on a different temperament. Afterwards, players were made to choose between which room they liked the most by going through the corresponding door. This forced the players to choose between two different temperaments, and thus making them express their preferences.

2.4.2 BrainHex

The work developed by Almeida [22] focused on creating a model for content placement to see if the order the content is presented in has any effect on the player's game experience. While developing his model, Bruno designed several challenges based on the different BrainHex [13] classes. These challenges made the player engage with different mechanics, depending on the BrainHex type they represented.

Afterwards, the challenges were presented to the players' in different orders, depending on their BrainHex questionnaire's results. He concluded that the order in which the challenges are presented may have a positive effect on the player's experience, which helps corroborate the idea that personalized content generation has a positive effect on the player's experience.

2.5 Discussion

We started this document by calling attention to the need for a precise way of identifying the player's preferences to create tailored experiences that better suit each individual. In this Related Work chapter,

we explored several aspects of our work by discussing the research we will use to achieve our goal.

We started by talking about personality type models in the field of psychology, discussing the MBTI and FFM models. This is an important section since it serves as the base of our work, where we assume that different individuals have different ways of interacting with the outside world, and as a consequence, will exhibit different behaviors while interacting with the game world. It is also very much essential to the work developed in player type models, which our work will make use of to identify a given player's preferences. This brings us to the second section of this chapter, Player Type Models, where we discussed several models, such as Bartle Taxonomy of Player Types, GMP, Marczewski's Player and User Types Hexad, and BrainHex. Our work will use the BrainHex model since its results are formed solely by the answers the player gives to the questionnaire, and it has a wide range (seven classes) of representation in regards to player preferences. The BrainHex model is constructed around seven different classes. However, since the Social class is inherently tied to deep interaction with other players, we will not worry about this class, considering our testing environment will consist of an offline, single-player game, where the preferences of a player with the Social class cannot be accurately represented.

With our work focusing on extrapolating the player's BrainHex class from their in-game behavior, we needed to discuss Data Mining techniques since we will need them to analyze the data and generate a model capable of performing such prediction. We started by talking about the tool we will use to run the machine learning algorithms, WEKA, followed by an overview of the machine learning algorithms' inner workings we feel we should try to test on our data. The algorithms we discussed were Decision Trees, K-Means, and Naive Bayes.

3

Case Study

Contents

This chapter will describe how we created a system that allows the design of simple goals to extrapolate the player's preferences for later use in data analysis. For this purpose, we will use the BrainHex player model, not considering its social component since the testbed game will be a single-player offline experience.

Our hypothesis states that we can derive the player's preferences by gathering data from meaningful game data-points. This data can then be used in future work to generate content tailored to the given player's preferences.

The topics that we will address in this section are the following:

-How, from the completion of different challenges, we can deduce the player's BrainHex types and how they correlate to each other.

-How can we design challenges/goals to target specific player personality types.

-Use the points discussed above to build an example game of how our hypothesis can apply to a real scenario.

-How we propose to validate the system with user testing using the scenario we built.

Our scenario will be, for simplicity, composed of two different modules that will run separately, but they could be adapted to work together at runtime. The first module is the scenario itself, where we will record the player's progress throughout the game, recording data related to the tasks they perform in the environment around them. The second module will use a third-party software, WEKA, to perform offline analysis of the data collected from the player. As stated above, these modules could be combined into one, to enable the use of the processed data to generate further content tailored to the player. Since we will not cover the generation of content tailored to the player, and our focus isn't to have an online system, we don't need to worry about that aspect, sticking with handling the telemetry data offline.

3.1 Testbed Game

To test our hypothesis, we developed a testbed game to collect players' data, which we will give the name of Legend of the Warrior's Crystals (LWC) fig. 3.1, with the entire map of the game accessible in appendix D. LWC is a single-player, 2D, top-down experience focused on allowing the collection of diverse player data. We chose to make a 2D top-down game since it allows for a more comfortable experience for the average player to engage in most game mechanics. The mechanics, although diverse, will be relatively simple for the same reason. Some parts of the game, however, will be more challenging, requiring the player to beat more complex and difficult challenges, even though these will be made by combining the more simple mechanics of the game. These options will allow us to use a broader range of possible game testers regardless of the gaming experience they possess.

The game was developed using the Unity game engine, with the help of a Unity Asset Store's as-

set called *"TopDown Engine"* [1] to significantly cut development efforts and allow a broader range of gameplay options.

Figure 3.1: Look and feel of the testbed game, as seen by the player.

3.2 Game Structure

The game is composed of two different sections, an "Outside" section where the player can explore the world and navigate through different paths, and an "Inside" section where the player can face specific challenges, both designed around the BrainHex classes. The full overview of the game's map can be seen in appendix D. At the start of the game, the player is shown a window with the controls of the game, both for keyboard and mouse, and gamepad. After closing the controls' window, the player is introduced to the game world via a quick interaction with an Non Playable Character (NPC), which gives them the main quest of the game: delivering a letter to his son, and introduces the player on how to pick up items from the floor and inspect them in the inventory. The player then follows a small path where they will pick up the sword they will use throughout the game. Using the sword they just picked up, they fight their first enemy. Afterwards the player arrives at the first intersection, finishing the "Tutorial" and entering the "Outside" section. After finishing the main portion of the game, the player arrives at the "Village", where they can interact with a couple of NPCs, which will guide them to the location of the NPC to whom they are delivering the letter to. Upon delivering the letter, the screen turns black, the game ends, and the

[1] https://topdown-engine.moremountains.com/

ending splash screen is shown.

The main quest's (delivering the letter) purpose is only to give the player motivation to arrive at the "Village", and does not serve to influence their decisions anywhere throughout the game. The player is free to navigate to anywhere they want to, and take on any challenge they might desire.

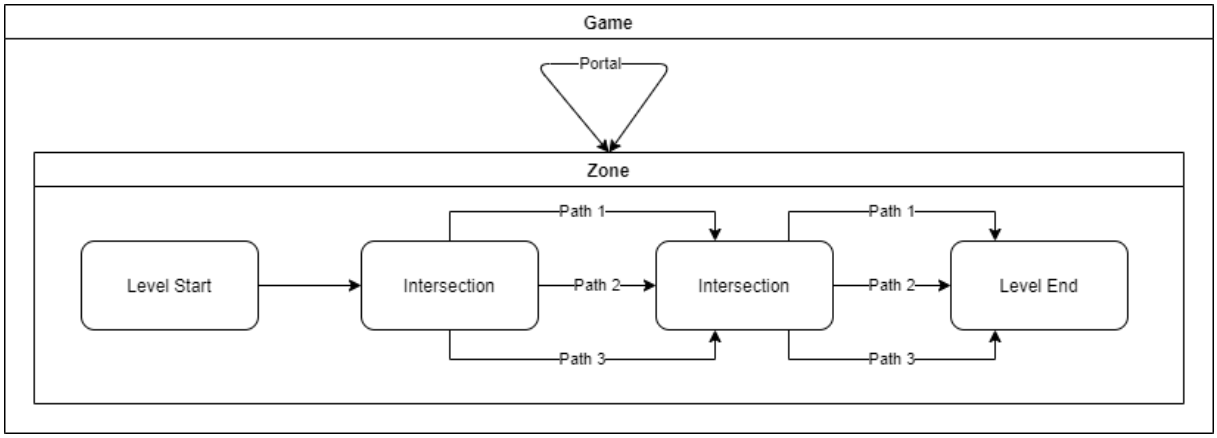


Figure 3.2: The structure of the "outside" section of the game.

3.2.1 Outside Section

As we can see in fig. 3.2 the game's "Outside" section is composed of different zones connected by portals. Each zones has four different elements: the "Level Start", the "Level End", the "Intersections" and the "Paths". The player always starts a given zone in the "Level Start", and are guided along a small path to the first intersection. At the intersection (fig. 3.3) the player will always find an NPC that gives a brief description of the three different paths the player can choose from. At the end of the zone the player is directed through a small path to a portal which will lead them to the next zone.

Each path in this section represents one of four different BrainHex classes: Mastermind, Conqueror, Daredevil, and Survivor. With the player being forced to choose between at least one of them at each intersection (section 3.2.3). The other two BrainHex classes, Achiever and Seeker, are represented by small quests, hidden areas, and game metrics (more on this later), instead of paths like the other four.

In the image, fig. 3.3, we can see an example of an intersection. In the middle, we see the NPC, that the player can interact with to obtain information regarding the paths ahead. The NPC will tell the player three different sentences, each for one of the three available paths. Every sentence is derived from the "I like to..." phrase of each BrainHex class available on the official website[2]. Every path also has a checkpoint right before the actual challenge to allow the player to respawn near their death place without having to move to the same spot again, from the beginning of the level. There will also be a

[2]https://blog.brainhex.com/

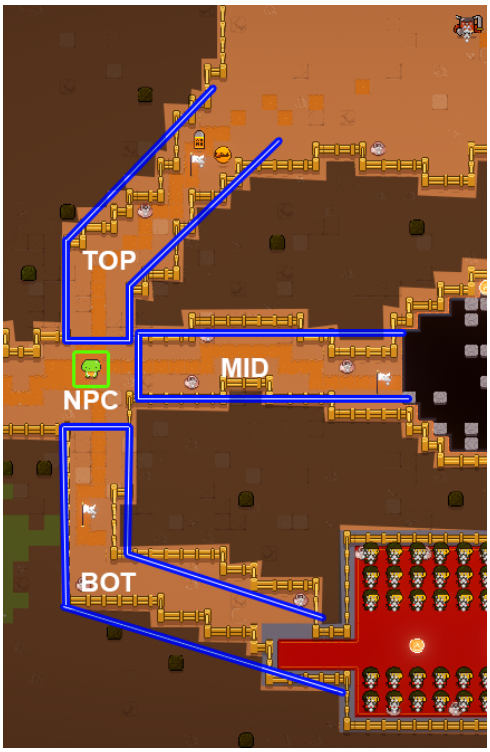


Figure 3.3: An example of an intersection splitting into three paths. This image is not representative of the in-game view of the player, as it is extremely zoomed out, does not have visibility effects that would hide certain areas of the terrain, and it grayed out in non-playable areas.

checkpoint flag right after the end of each intersection so that if the player completes one of the paths, they won't have to redo them upon death later on.

In order to give the player time to experiment with different aspects of the game, and not skew our initial measurements, we created, right after the "Tutorial", a small zone where the player can easily experiment with the different mechanics and types of challenges the game has to offer. This section starts with the first intersection the player finds, and spreads until the end of the first zone, where the player takes the portal to zone two. With a small pathway with Survivor style mechanics and challenges, a small path with enemies, hidden zones the player can explore, small jumping challenges which require precision, and a couple of pots and coins for the player to collect, every BrainHex class is represented. Upon arriving to the second zone, the player will have already experienced a bit of what every BrainHex class has to offer and will probably start choosing according to their preferences, instead of choosing for experimentation, and we call this zones from here on out the main portion of the "Outside".

Regarding the intersections present in the main portion of the "Outside", we chose to have four of them, each with three choices the player can choose from. Since paths are only created on the basis of the classes Conqueror (M), Daredevil (D), Survivor (S) and Mastermind (M), and to make sure we have a good distribution we decided to always split them vertically with one top path, one middle path and one

bottom path, and ensure we don't have the same choice in the same place two times. This can be seen in table 3.1. This gives the player the ability to choose their favorite type three out of four times, but also make sure they choose at least one other path type, giving us more information about their preferences. Having three choices per intersection helps to not overwhelm the player with choices, but also increase our chances that they will like at least one of the paths presented, which means they are not forced to do something they are not found of.

Intersections	Top Path	Middle Path	Bottom Path
1	Daredevil	Mastermind	Conqueror
2	Survivor	Conqueror	Mastermind
3	Mastermind	Survivor	Daredevil
4	Conqueror	Daredevil	Survivor

Table 3.1: Table showing the distribution of the different path types across the different intersections.

3.2.2 Inside Section

The "Inside" sections of the game are connected to the "Outside" sections via a bidirectional portal located somewhere in the "Outside Zone's" as seen in fig. 3.4. Each "Inside" sections, or Challenges, are completely optional for the player (section 3.2.3). Similarly to the intersections, the player is given a brief description of the type of challenge behind the portal via an NPC located near the portal in the "Outside Zone". Upon entering a challenge another NPC will provide the player with a bit of information on the mechanics of the challenge if needed. When the player reaches the end of challenge they will be rewarded with a chest containing an assortment of items for either collection or gameplay use, such as coins and health packs, respectively. Upon collecting the final reward the player is presented with a portal that returns them to the same place in the "Outside" section they entered from.

3.2.3 Player Freedom

The main objective of this work is to see if it is possible to measure the player's preferences through the behaviour they express while playing the game. In order to achieve this it is of the utmost importance to give the player the freedom to engage with whichever elements of the game they want to. This means only forcing the player to complete one path per intersection, with all other elements being completely optional. This means the player, apart from completing one path per intersection can choose to:

- Give up mid-way through any path, go back to the intersection, and start a different path.
- Complete more than one path from each intersection (even all of them).
- Engage with zero Challenges ("Inside" sections) if they want to.

- Enter a Challenge and give up mid-way through it.
- Complete a given Challenge more than once.
- Explore the entire map, or stick solely to intersections' paths.
- Collect every item they find, or not collect anything at all.

This game design allows the player to explore and engage with the game however they feel more comfortable doing so, hopefully expressing their gaming preferences in line with the BrainHex player model.

3.3 BrainHex Classes In The Game

Our objective is to determine the player's gameplay preferences from the way they engage with our game. To achieve this we need to create a foundation from where the player is able to express their preferences while playing the game (section 3.2), which requires giving them the freedom to engage with it as they prefer the most, but also making sure they are making meaningful choices (section 3.2.3). This means that the players are provided with an environment where their choices contribute in some way to the identification of their BrainHex classes.

With this in mind, we decided that the player should have the freedom to choose to not engage with content that they might not want to (section 3.2.3), as an example: a player that does not like Mastermind type content should not be forced to play it. However, a player needs to have a minimal level of engagement with the game to be able to properly express their preferences.

In order to design a game where the player is able to properly express their preferences we need to start by creating content which has some kind of correlation to the different types of preferences we are trying to measure. In our case this means the creation of content which is directly correlated to the six different BrainHex classes we are trying working with. In light of this we decided to start by creating six different challenges, each one corresponding to one of the six BrainHex classes. This lead to the creation of two different types of challenges: the challenges constrained to a certain space (the entire challenge takes place in a closed off room, separated from the rest of the game, as seen in fig. 3.4), and open type challenges, that occur in a closed off room, but also need the player to perform actions in the outside world.

The differentiation between closed off and open type challenges was done on the basis that different BrainHex classes require different types of challenges. The Mastermind, Survivor, Conqueror, and Daredevil classes can be accurately designed in a closed off room challenge, while the Seeker and Achiever classes require a more continuous and open ended expression of the player's behaviour by their definitions.

Figure 3.4: The mastermind closed off challenge.

Having said that, we decided to focus on four of the six BrainHex classes being evaluated (Mastermind, Conqueror, Daredevil, and Survivor) which can be properly represented as standalone challenges in paths. This is not achievable for the other two classes, where there exists a greater need to examine the overall behaviour of the player to determine their relatedness to the given class. From this perspective we created mini-quests, hidden areas and game metrics (coins,etc..) to represent the Achiever and Seeker classes.

With all these in mind, in order to properly design paths and challenges representing each BrainHex class, we looked to the official BrainHex website[3], and took into account all the information provided for each BrainHex class, such as what someone who identifies as the given class likes, how they usually behave, their favorite types of games, and their class's relation to other player and personality models. As such, we came up with the following overall descriptions of what should be included in the different types of challenges, paths, and metrics according to their respective BrainHex class:

Seeker: The player will need to search for a key to a hidden door that can only be obtained by thoroughly searching the level. The player will find strange and wonderful scenarios.

Survivor: The player will need to survive trapdoors, spikes, monsters, and other elements that may be deemed as "scary".

Daredevil: The player will need to overcome a challenge filled with moving platforms, trapdoors, and

[3]https://blog.brainhex.com/

be very precise with their timing.

Mastermind: The player will need to use limited resources to solve a puzzle with moving objects and pressure plates, among other things.

Conqueror: The player will need to overcome a series of difficult enemies with increasing difficulty.

Achiever: The player will try to do everything available to them in the game.

Some examples of challenges implemented into the game using the above descriptions, and concepts discussed before are:

Figure 3.5: An image of a part of the Seeker challenge, where the player is rewarded with a scenery and a spectacle by the blue creature in the lake.

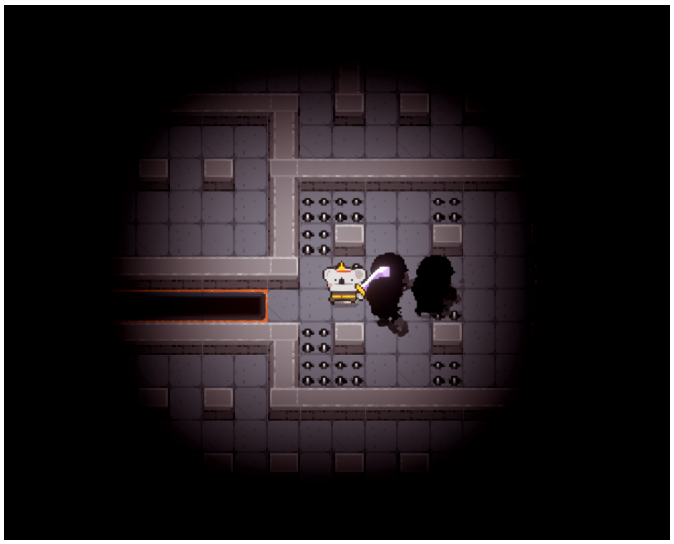


Figure 3.6: An image of a part of one Survivor path, where the player running away from ghosts in a room, in a room filled with traps, while under the effect of a limited view radius.

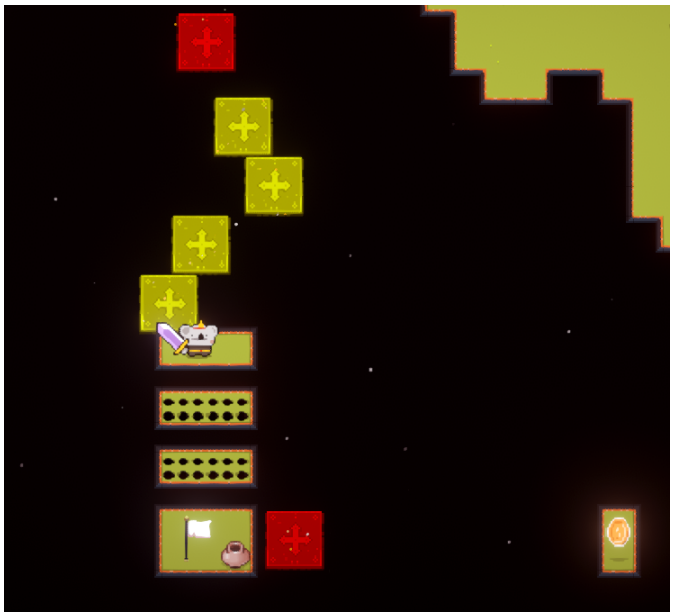


Figure 3.7: An image of a part of one Daredevil path, where the player running away from ghosts in a room, in a room filled with traps, while under the effect of a limited view radius.

3.3.1 Seeker

The Seeker challenge depicted in fig. 3.5 and video[4], makes the player find three green coins scattered through one of the "Outside" levels. Upon delivering the three coins to the NPC, a portal opens up and the player finds themselves in the environment in the picture, where they are rewarded with a view of the scenery depicted in fig. 3.5, and a spectacle made by the blue creature we can see in the lake. The foxes to the right of the player will only show up if the player managed to find the hidden place where they were hidden in the "Outside" level, providing another layer of engagement with the seeking mechanic, giving a familiar view as a reward, much like in the description of the Seeker class.

3.3.2 Survivor

In the Survivor path depicted in fig. 3.6 and video[5], the player is navigating a closed off, claustrophobic ruins, where they will need to run away from ghost enemies, explore hidden paths, and make sure they don't get caught by traps while under the effect of the view restriction seen in fig. 3.6, with a creepy song playing in the background.

[4] https://youtu.be/XCHGVLS8q0I
[5] https://youtu.be/mYy47ONc4Eg

Figure 3.8: An image of a part of one Mastermind path, where the player is given a riddle via the scroll shown in the left of the screen, and needs to solve it by stepping in the nine blocks in a grid shown in the picture.

Figure 3.9: An image of a part of one Conqueror path, where the player is fighting a strong enemy.

3.3.3 Daredevil

In the Daredevil path depicted in fig. 3.7 and video[6], the player is jumping across moving platforms, timing their movement with the spikes that come up from the floor, while trying to avoid falling out. To navigate the platforms the player needs to have quick a precise movement, like explained in the Daredevil BrainHex class description.

3.3.4 Mastermind

In the Mastermind path depicted in fig. 3.8 and video[7], the player is solving a riddle given to them in the form of the scroll illustrated in the left side of the picture. They are required to step on top of the nine blocks in grid shape in the correct order. Upon completion, the big block on the right disappears and the path is completed.

[6] https://youtu.be/Erdug9eO-K4
[7] https://youtu.be/lGxheUwJWjs

Figure 3.10: An image of a part of the Achiever challenge, where the player is required to collect 200 pots.

3.3.5 Conqueror

In the Conqueror path depicted in fig. 3.9 and the video[8], the player is fighting a strong enemy, which has a bigger health pool than the player, moves faster, and takes away a lot of player health with each strike. This proves to be a hard challenge for the player to overcome.

3.3.6 Achiever

The Achiever challenge depicted in fig. 3.5 and video[9], makes the player collect 200 pots before the NPC gives the key needed to open the door to the chest. In this challenge the player needs to venture to the "Outside", and collect pots throughout the entire game. Upon collecting all the necessary pots, the player can use the teleport stone given to them by the NPC to quickly return to the area of the challenge and complete the challenge.

3.4 Manipulation Check

Before moving on to the final test we had to verify that the challenges we designed actually belonged to a given BrainHex class. To achieve this, we presented some users with six videos [10] [11] [12] [13] [14] [15]

[8] https://youtu.be/IVHTWJKYtIU
[9] https://youtu.be/GYxu0mKxyjo
[10] https://youtu.be/IVHTWJKYtIU
[11] https://youtu.be/Erdug9eO-K4
[12] https://youtu.be/lGxheUwJWjs
[13] https://youtu.be/mYy47ONc4Eg
[14] https://youtu.be/XCHGVLS8q0I
[15] https://youtu.be/GYxu0mKxyjo

(one for each BrainHex type), showcasing our game design philosophy for each BrainHex class. The questionnaire used can be seen in appendix B.

3.4.1 Characterization

Our experiment collected data from nine individuals, from which 2 were female and 7 were male, with ages ranging from 18 to 48 years old, with a mean of 24.77 and a standard deviation of 8.88. 77.8% of users said they make some time in their schedule to play video games, while 22.2% said they play video games occasionally when the opportunity presents itself. 44.4% of users said they "enjoy and have played/watched others play, multiple times, games in which the protagonist combats a large number of enemies by shooting at them while dodging their fire", while 55.6% said they "played/watched others play them enough to understand they do not appreciate them". This last question ensured that an individuals fondness for the type of game we are evaluating didn't affect their understanding of the challenge.

3.4.2 Challenge Validation Questionnaire

To acquire the data needed to verify our design for the challenges we used a questionnaire format, where the user needs to rate each of the six videos in relation to six provided sentences. Each video resembles one of the six different game design philosophies we employed while making the game, described by each inside challenge we created for the different BrainHex classes (appendix B). The questions were presented in a table, with the rating as columns and the sentences to describe each of the six BrainHex classes as rows. The table was displayed after each video for the user to fill out. The rating for each challenge ranged from "Strongly Disagree" to "Strongly Agree", with the following options in between: "Disagree", "Slightly Disagree", "Neutral", "Slightly Agree", and "Agree". The BrainHex sentences were taken from the official website and represent what each player identified as belonging to a specific class likes to do.

- Daredevil - "You like negotiating dizzying platforms or rushing around at high speed while you are still in control."
- Conqueror - "You like defeating impossibly difficult foes, struggling until you eventually achieve victory."
- Mastermind - "You like solving puzzles and devising strategies."
- Seeker - "You like finding strange and wonderful things, or finding familiar things."
- Survivor - "You like escaping from hideous and scary threats, pulse-pounding risks."
- Achiever - "You like collecting anything you can collect, and doing everything you possibly can."

3.4.3 Results

Since we allow the users to rate each sentence individually in accordance to how much they thought it related to the video shown, the need aroused to come up with a way of validating the challenge with the design intent in mind. With the fact that the challenges presented were all created on top of the same, 2D run and gun base game, we expected some overlap in identifying the correct BrainHex class. With this in mind, we settled for accepting a positive answer if the user's score for the correct class' sentence is greater than the score for all other sentences. As an example if for the video of the Mastermind challenge, the user rates the Mastermind sentence with the highest score of all presented options, then we accept that answer as positive. On top of this, we also observed if the score for the correct sentence was at least "Slightly Agree", to make sure the user didn't just rate the challenge as low for all categories, which is also not a desirable outcome.

With the corresponding ratings going from 1 ("Strongly Disagree") to 7 ("Strongly Agree"), all six graphs with the average ratings given by all participants to each video for each BrainHex class' sentence are: fig. 3.11.

The results also showed, as we expected, that most challenges also had some connection to more BrainHex classes other than the main one we were trying to represent. This is attributed to the fact that challenges in a somewhat complex game can't really be created based on a single unblended, isolated class, as there will always be some residual components that can be associated with other BrainHex classes. We argue that as long as the main BrainHex class is identified as the single most important component of the challenge, then it can be used as a challenge for the given class in our game. These may, later on, also help in identifying oddities with our models or data.

With the data from this questionnaire, we ended up changing some challenges to better isolate the intended BrainHex class. The changes performed were:

- On the Survivor challenge, the enemies had their health bar removed and were made immortal to incentivize the player to run away from them instead of thinking they were supposed to fight them. This helped lower the Conqueror component of the challenge.
- Coin collection was mostly removed from the Survivor and Daredevil challenges, which helped lower the Achiever component.

With a game scope as large as ours it was not feasible to make pass through the manipulation check procedure all the challenges we designed. With this in mind, and with the main challenges of the game already validated, we decided to create the rest of the mini-quests, hidden areas, and paths, based on these validated challenges and the overall ideas on which we based them, which we discussed in section 3.3.

Although the results presented in fig. 3.11 show us that, on average, participants correctly identified

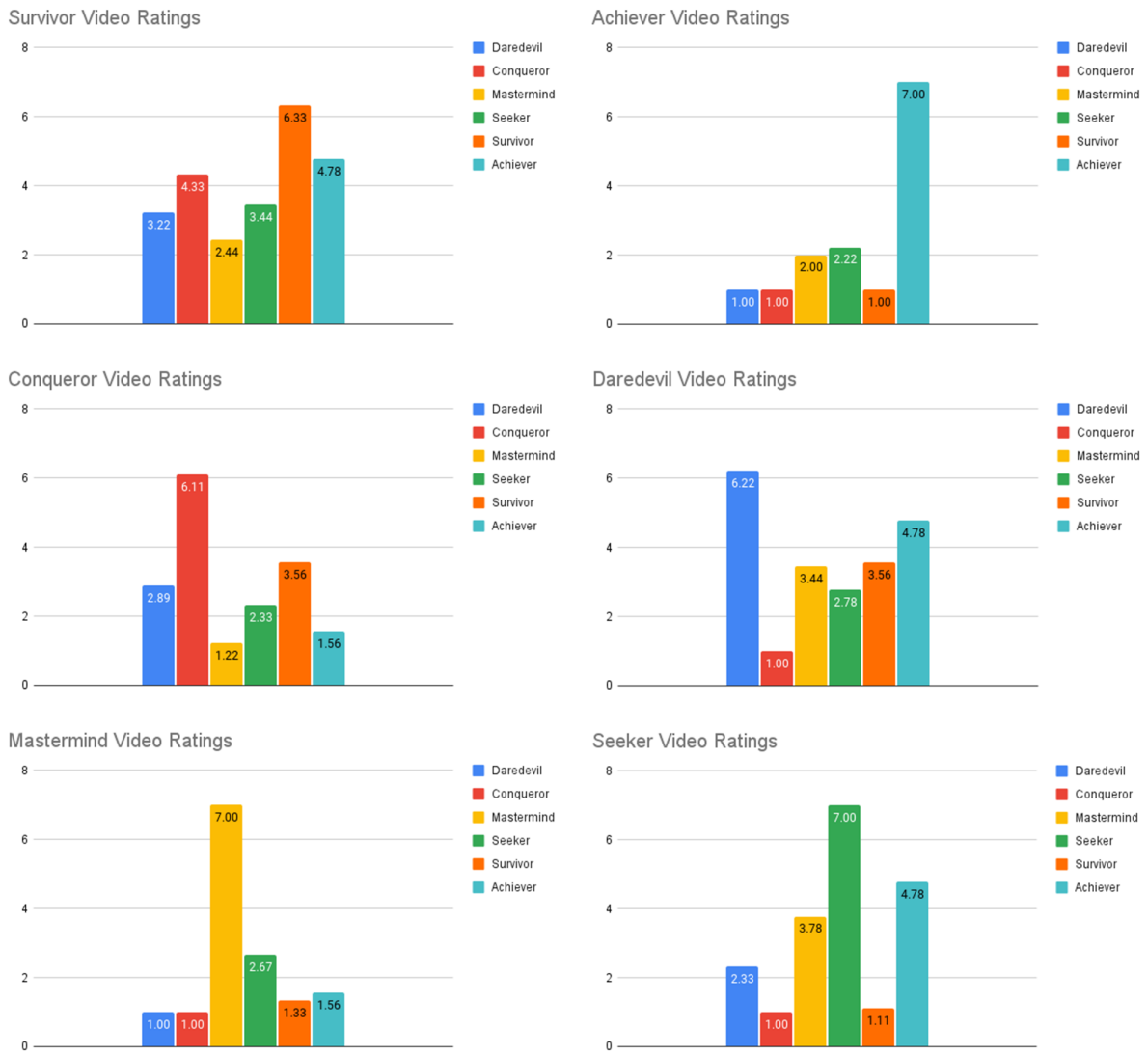


Figure 3.11: The average rating obtained for each of the six classes' videos in the manipulation check questionnaire.

the BrainHex class, the individual results also showed us that everyone attributed the highest score for each video to the correct class. There was, however, one exception with one participant incorrectly scoring all challenges. We considered this result an outlier, since there were clear indications that the questionnaire had been answered randomly.

3.5 Game Data Collection

In order to perform data analysis and apply machine learning techniques we first need to collect relevant data from the players' in-game behaviour. Since we cannot be sure of what might actually end up being

relevant data to the experiment, we decided to log every relevant action the player performs inside the game. This allows us to, if needed, reconstruct the entire play session from the logged game data. We decided to draw a line on what is considered relevant in order to not overload our logged data with irrelevant information. This meant that key presses, mouse movements, or game-pad actions were not recorded, however, every action the player character performs in-game is registered. With this in mind, we recorded the following actions taken by the player character:

- Picked up items, such as coins, health packs, and guns, among others.
- Defeated enemies, such as simple ninjas, bosses, and ghosts.
- Areas entered and exited from, such as intersections, paths, hidden places, and zones, among others.
- Deaths, with where and how they happened.
- Checkpoints reached, and respawn locations triggered.
- Quests started and completed.
- Objects interacted with, such as levers and chests.

On top of all the information described above, we also needed to acquire information on how much the player progressed into a certain path of challenge if they didn't finish it. This information is needed so that, if the player gives up mid-way through a path or challenge, we can score how much they engaged with it. In order to do this, we designed a system where we could use the information from the areas entered and exited, as described above, to initiate a separate logging system, which would write to the same log file, but with a different prefix in each line, to make it easier for the parsing script. In order to know how much the player engaged with a given path or challenge, we made use of the information in the list above to create an event system, which would progress as the player reached predefined parts of the challenge/path, defeated a certain number of enemies, or collected specific items, adding to their score. Although this system required us to specify every possible action that could contribute to the player gaining score in the challenge/path, it also provided fine-grained control over how the player is scored.

Some examples of metrics used to attribute a score to each of the BrainHex classes' challenges, paths or quests are:

- Seeker Challenge: This challenge requires the player to search for hidden coins throughout one Level, delivering them to a NPC and experiencing a small scenario with a fantastical creature. The progression is first measured by how many coins the player managed to find, with each coin having a separate value depending on how hard they are to discover, capping at 0.45 out of 1.

The remaining 0.55 of the score is distributed by the delivering of the coins to the NPC, and the experiencing of the wonderful scenario.

- Conqueror Challenge: This challenge requires the player to fight 3 waves of enemies inside a closed off arena, with a final Boss at the end. The score is distributed by how many enemies the player manages to defeat, with a bonus for finishing each wave.
- Achiever Challenge: This challenge requires the player to find and collect 200 pots throughout the entire game. The score is distributed to the player in relation to how many pots they collected up to a maximum of 0.9. The final 0.1 of the score is only given after they deliver the pots to the NPC.
- Survivor Challenge: This challenge requires the player to find their way through a small cavern filled with traps, ghost enemies they cannot fight against, with a limited range of vision and while listening to a creepy song playing in the background. The score is attributed to the player based on how far along the cavern they managed to get. As with many other challenges, if the player fails and tries again, without ever succeeding, they will receive a small bonus points if their total score does not surpass 0.9.
- Daredevil Challenge: This challenge requires the player to jump across moving platforms, time their movement with the spikes that come up from the floor, while trying to avoid falling out. This requires precise movement and for the player to keep some speed to not fall behind. The score is attributed based on how far along the path they managed to get.
- Mastermind Challenge: The mastermind challenges all consist of different puzzles with different mechanics, which means they all have different scoring systems. If the path has multiple puzzles, each puzzle will be graded separately and the combined score is always equal to 1. Some puzzles do not have a way to measure the player's progress so they either award the full score, or zero.

All actions were timestamped, which allows us to know, if we need it, how long the player took to complete any action, quest, or challenge. This metric was discarded later on, since we were made aware by several testers that since the experiment could take more than one hour, some of them decided to take breaks in between, which might change the result. This meant we didn't have concrete mechanisms to do this in an absolutely rigorous way, which could be tackled in future works. This leaves us with a log file composed of separate lines, where each one corresponds to one action taken by the player character, timestamped with the number of seconds elapsed since the game was started.

Data Retrieval

In order to gather the logged gameplay data remotely from the user, a system to send it back to us was needed. Several solutions were explored, such as: the uploading of the log file to the google forms questionnaire by the user, which was discarded since it had a lot of points of failure like the user

forgetting to upload the file, and the need to have a google account to perform the action, the other option was to automatically upload the file to the cloud, streamlining the process on the testers' end and assuring that the file is delivered. The second option was the chosen one, and a system to automatically send an email with the log file as an attachment to an email generated for this specific purpose was put in place. This means that upon triggering the end screen, in the background, an email is sent with the attached log file and the corresponding Unique Identifier (UID) given by the tester. The tester is informed about this, happening, before, its occurrence, and can opt-out of the experiment if they choose to do so. Upon receiving the log file with the data, it was matched with BrainHex results, and the final questionnaire answers using the UID.

3.6 The Final Experiment

The experiment was divided into three parts: the pre-game questionnaires, playing the game, and the post-game questionnaire. With the portions of the three parts taking place in different platforms and programs, a need to anonymously identify the participants arose. To solve this problem we decided to generate a random UID, which will identify the participant in all questionnaires, including the BrainHex results, and game log data. The participant will be asked to input this UID in all questionnaires, and in a dialogue box upon starting the game. The experiment can take anywhere between 30 and 90 minutes depending on the play-style and choices of the player while playing the game.

3.6.1 Pre-Game Questionnaires

There are two different questionnaires the user will have to answer before playing the game. The first is a demographic questionnaire, and the second is the BrainHex player model questionnaire, both can be seen in appendix C.

The demographic questionnaire is hosted through google forms, while the BrainHex player model questionnaire had to be custom made for the experiment. This happened because we felt that the player should not have access to their results of the BrainHex questionnaire before finishing the experiment as it may influence their decision making and overall behaviour while playing the game. In an ideal scenario, the player would answer the BrainHex questionnaire a couple weeks before playing the game, so as to not influence their behaviour throughout the rest of the experiment. This proved to not be feasible in this time constrained work, where getting a relevant number of players for statistical significance is already hard, and asking people to do a questionnaire, wait a couple weeks and reengage with the experiment might prove to be too difficult. To solve this, and in collaboration with another colleague, we created a custom made website[16], which can also be seen in appendix A, hosted in Técnico's servers, where the

[16] https://web.ist.utl.pt/ ist186383/

players can answer the BrainHex questionnaire, and at the end, we generate a random UID and give it to the user to track them anonymously through the rest of the experiment. The UID is then stored with their BrainHex results in a text file, which is not available to the user.

At the start of the experiment, the player is sent a link to a google forms questionnaire where they will be asked some demographic questions like their age, gender, how often they play video games, and how familiar are they with the game genre of our game, where the protagonist combats a large number of enemies by shooting at them while dodging their fire.

Afterwards, the questionnaire redirects the user to the purpose built website for the BrainHex player model questionnaire, at the end of which the user is given their UID to copy and paste into the demographic questionnaire. The user is now finished with the BrainHex website and can close it.

Upon returning to the google forms questionnaire, the player is asked to enter their UID. After doing that, the user is given a link to download the game from. The google forms questionnaire is not closed.

3.6.2 Playing The Game

After downloading and starting the game, the player is presented with a text box where they are asked to input their UID into. Only after completing this step is the play button made available to them. After playing through the game, the logged game data is automatically sent in the background with the attached UID. The player is then asked to return to the google forms questionnaire.

3.6.3 Post-Game Questionnaires

Upon finishing the game, the player is presented, in the google forms questionnaire, with the in-game module of the Game Experience Questionnaire (GEQ) [23]. This part was designed to evoke the participant's feelings and thoughts while they were playing the game. This serves to indicate any problems that may arise from the experiment, namely about the in-game experience of the player, like if the player was bored while playing the game, if they felt frustrated, if they felt skillful, among others.

3.7 Discussion

In this chapter we started by discussing the structure of the game, followed by the different types of metrics we collected from the players behavior in the game, then we talked about the steps taken to ensure the content we created was appropriate for our purpose, via the manipulation check, and finished with the description of our final experimental procedure. All the steps outlined in this chapter will help the reader in the replication or improvement of this study.

4

Results

Contents

This chapter will describe how we treated the data for use with our machine learning software, WEKA, and its various algorithms, and how we then worked with the data to ensure we get the best possible results with our models' predictions.

The topics that we will address in this section are the following:

-The demographics of the users that participated in the final study.

-How the data gathered from the users' playtesting was parsed to better represent, in a concise and meaningfull way, their actions and choices inside the game.

-How the data was converted to a format usable by WEKA.

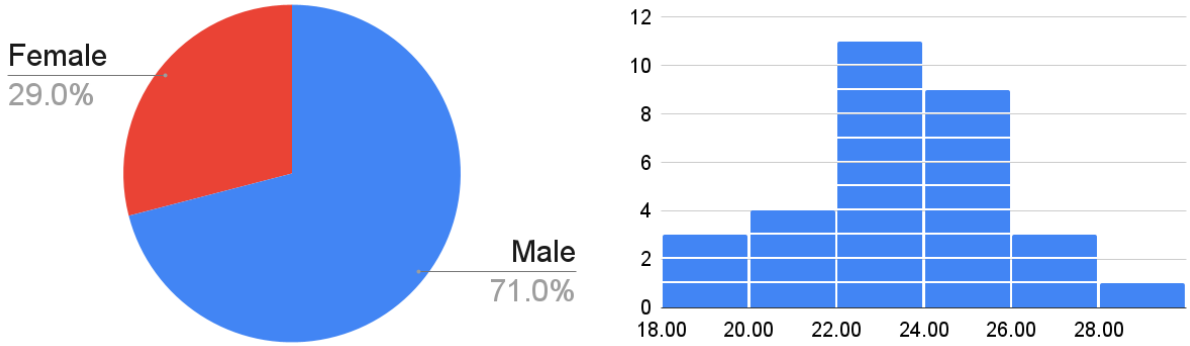
-The process of creating the different final machine learning models, the algorithms explored, and how the data was processed.

-The final results from the experiment, and their analysis.

4.1 Demographic Results

The demographic of the experiment's testers was mostly from people with a gaming background, from a young age group. The link to the google forms questionnaire which started the experiment was distributed among students of Instituto Superior Técnico (IST) via convenience sampling, and other people contacted personally. In total, we got 30 users participating in the experiment, with a 100% completion rate.

Our experiment collected data from 30 individuals, from which 29.0% were female and 71.0% were male (fig. 4.1 graph (a)), with ages ranging from 18 to 29 years old, with a mean of 23.03 and a standard deviation of 2.30 as seen in fig. 4.1 graph (b).



(a) Gender **(b)** Age

Figure 4.1: Pie chart and histogram with the gender and age information of the 30 testers, respectively.

Regarding our testers gaming background, 64.5% of users said they make some time in their sched-

ule to play video games, while 35.5% said they play video games occasionally when the opportunity presents itself as seen in fig. 4.2.

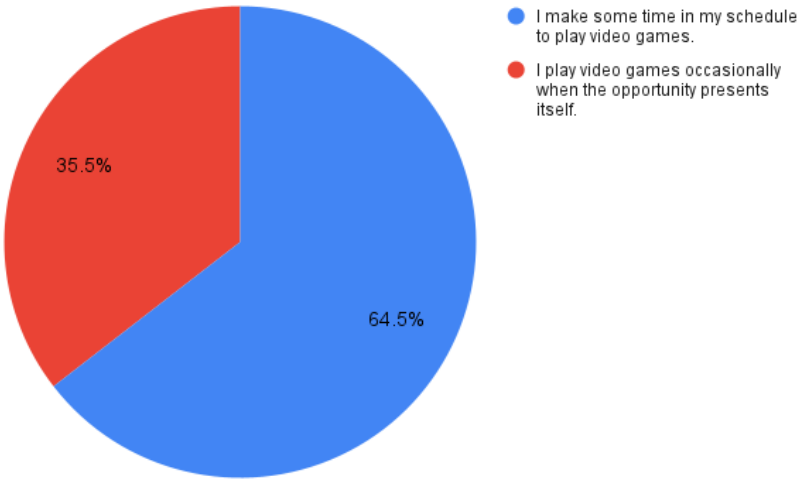


Figure 4.2: Pie chart showing the time participants spend playing games.

With respect to our users familiarity with our game's genre, and as seen in fig. 4.3, 45.2% of users said they "enjoy and have played/watched others play, multiple times, games in which the protagonist combats a large number of enemies by shooting at them while dodging their fire", 38.7% said they "played/watched others play them enough to understand they do not appreciate them", and 16.1% said they are not familiar with these games and/or have no formed opinion on them.

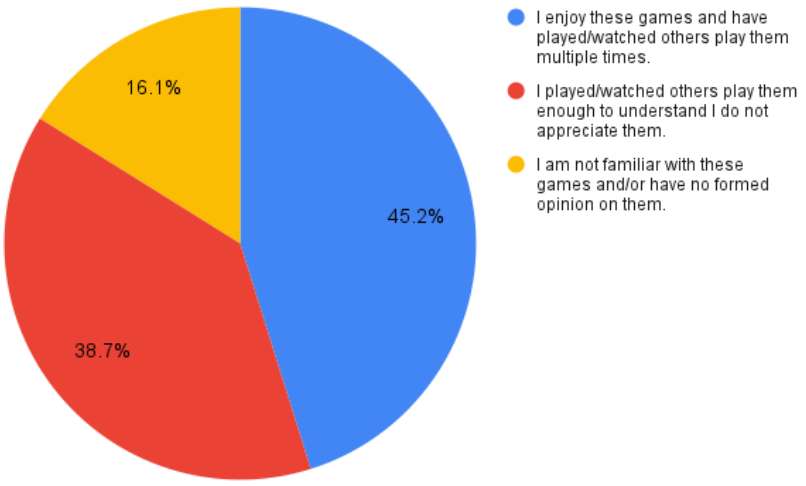


Figure 4.3: Pie chart showing how familiar the participants were with the game's genre.

Regarding the results from the BrainHex questionnaire, table 4.1 presents the distribution of positive and negative values recorded for each BrainHex class. A positive value corresponds to a BrainHex score

in the range [10,20], while a negative score represents a score in the range [-10,9].

	Seeker	Survivor	Conqueror	Daredevil	Mastermind	Achiever
Positive	16	22	16	17	16	18
Negative	14	8	14	13	14	12

Table 4.1: BrainHex score counts for all participant. Positive values indicate a score greater than or equal to 10, while negative values indicate a score lower than 10.

As we can see, our coverage of the BrainHex space is pretty diverse, with only Survivor having a slightly more unsymmetrical representation. This helps in the differentiation of the data points, since if we only had one or two data points representing a given class, our results would have been severely skewed.

4.2 Data Treatment

4.2.1 Parsing

The data gathered comes in a scattered format as discussed in section 3.5, and needs to be properly organized to be used in WEKA. This process is performed by an automated script that looks at all entries in the log file and categorizes them, extracts information on what the player did and what they didn't do, and, if needed, attributes a score to their engagement with a given section of the game. The categorization takes the individual events and organizes them into two different types:

- Area events, which record if a given area was reached by the player, and how long they stayed there.
- Instantaneous events, like the player's deaths, kills, picked up items, among others. These are associated with area events.

Afterwards, the script parses through the collected information to attribute a score to the events above and categorizes them into the following variable types:

- Quest completion rates. If the player accepted a quest, and if they completed it. In case the quest was accepted but not completed, a score is attributed based on how far along the player reached into the quest. This is done based on predefined metrics.
- Challenges completion rates. Like quest completion, they mark if the challenge was completed or not, and in case it was started but not completed a score is also given to the players engagement.
- Path completion rates. Like the other two completion rates, this tell us if the player started a path or not, and if they gave up mid-way a score is given for how far along they were able to reach.

- Number of coins collected.
- Number of suits of armor purchased.
- Number of pots collected.
- Number of hidden areas found.
- Enemies defeated.

In order to ensure proper values/scores for the variables described above, we needed to come up with ways of normalizing the data for use within WEKA. To normalize the variables, our first idea was to count up the maximum number for each variable inside the game and use it as a ceiling for normalization. This, however, came with a problem for the coins and pots collected, and enemies defeated, since they will be inside repeatable challenges, making it impossible to get ceiling value for them. To fix this we decided to take the highest number present in the entire dataset as the ceiling value. This comes with the problem that if we decided to extend the dataset we would need to run it all through the script again, but it was a compromise we were willing to accept for our work. The path and challenge scores were already attributed with values ranging between zero and one, so we have no need to normalize them.

After the script finished parsing the data, we are left with a Comma-Separated Values (CSV) file, where in each column we have variables, and in each row the values obtained by each participant. A cropped version of one of these files can be seen in fig. 4.4. All values presented in the file are normalized for ease of use with the different machine learning methods available in WEKA.

The choice of using a CSV file format is motivated by readability, versatility of use with several different softwares, like Microsoft Excel [24], and WEKA, and ease of creation with our script. A typical CSV file is composed of several lines, each representing one instance, with elements in each line separated by a comma, and columns represent different variables. The first line of the file is, in our case, reserved for labeling the different values we use, like a header. A typical CSV file can be seen in fig. 4.4.

```
id,achiever_challenge,survivor_challenge,
60cb70f8830fc,1,0,1,1,0,0,0.25,0,0,1,0,0,
60cb22c163475,1,1,1,0,1,1,1,1,0,1,1,1,1,0
60bd100087519,1,1,0,0,1,0,0,1,0,0,1,1,0,1
60bcdb09d167e,1,0,0,1,0,0,0.25,1,0,0,1,0,
60bcc489428b5,0,0,1,0,0,1,1,0,1,0,0,1,0,0
60cf26e2a9fd0,0,0,0,0,0,0,0,1,0.25,1,0,0,1,
60df32a8ad44a,0,0,0,0,1,0,1,0,0,0,1,0.15,
60e0377e54d87,1,1,1,0.35,1,0,1,1,1,1,1,1,
60e073be83009,1,1,1,1,0,0.65,1,1,1,1,1,1,
```

Figure 4.4: Example of a cropped part of one of the CSV files generated.

4.2.2 Spliting The Data

With the data parsed to a format usable by WEKA, we switched our focus to how we should handle the data for training our models. Taking into account that our dataset has a small number of samples (n=30), if we were to randomly split the dataset into training and testing sets, we might run into a problem of having all "positive" results fall into one of the sets, while the other ends up with none, or vice-versa. This means that we need to make sure both the training and testing sets have the same proportion of positive-negative instances, however another problems arises, which is how to define a positive or a negative instance. To handle this we had two options:

- Option 1: Do a simple dataset split into 70% for the training set, and 30% for the testing set, leaving the testing data untouched until the end of the process.
- Option 2: Make six copies of the dataset, one for each BrainHex class, and split the data into 70% for the training set and 30% for the testing set separately.

The two options will lead to different workloads regarding the training of our models and give us different options into how we can better prepare the dataset for each training process. With option 1, we would have to mark the instances separately, with each one having a different class depending on which BrainHex class gets the highest score. This leads to a couple of problems, namely the fact that the instance might have two BrainHex scores with the same value, and we would have to choose one, or have two scores with very high values, and all the others with low values, which might skew the training of the model, or the instance might have all scores very close to each other, which might also skew the training of the model. With option 2, we can treat each BrainHex class separately, and independently, which lets us have finer control over how we classify the dataset. We can make a score cutout for each BrainHex class separately, and classify each instance as belonging to the corresponding BrainHex class or not. This would also make it easier for evenly splitting the data for the training and testing sets. With this in mind we went with option 2, and created 6 copies of the dataset. Each copy was treated as a different dataset, and we settled for a cutout of 10 for the positive and negative values (table 4.2). This means that, for example, for the Mastermind copy of the dataset, we looked at the BrainHex scores of each instance and attributed a positive class (belongs to the BrainHex class), denominated by the value "1", if the instance had a BrainHex score higher or equal to 10, and a negative class (does not belong to the BrainHex class), denominated by the value "0", otherwise, as seen in table 4.2. The cut-off value of 10 was chosen since it marks the middle point in the BrainHex scale for someone who likes the given BrainHex class. The BrainHex scores vary between [-10,20], with values in the range [-10,0[corresponding to the player having the given class as an exception. We are then left with the value range [0,20] for how much the player likes the challenge and we chose the value 10 as it marks the middle point of how much someone enjoys that specific type of content.

Mastermind Brain-Hex Score	Class
10	1
9	0
16	1

Table 4.2: Example of a BrainHex score's discretization.

With this established, the final dataset looked something like table 4.3, where each column represents one of the variables treated in section 4.2, and each row represents one run made by a participant. The final column ("Class") represents, for the given dataset, if the participant scored a value higher than or equal to 10 in their BrainHex questionnaire, as discussed at the beginning of this section (table 4.2).

Mastermind Score	...	Path 1	...	Class
0.4	...	0.8	...	1
0.0	...	0.9	...	0
0.1	...	0.35	...	1
...	...	...	...	...

Table 4.3: Example of one of the six final datasets' structure.

4.3 Feature Selection

The filtering of our data occured right after importing each dataset into our machine learning software, WEKA. As the first measure we decided to check what WEKA's tools for feature selection could tell us about our data, for each BrainHex class's dataset we ran the *Best First* [1] algorithm with a search termination of five, using the *Correlation-based Feature Subset Selection for Machine Learning* [2] [25] attribute evaluator. This process already gave us a glimpse into the meaningfull data-points we could expect from each data set, and was performed using 10-fold cross validation. As discussed in section 4.2.2, this dataset consists solely of training data (70% of the original dataset), while the testing data (the remaining 30% of the original dataset) is not touched upon.

4.3.1 Conqueror

The Conqueror dataset, consists of all the variables discussed in section 4.2.1, along with the class indicating if the participant had a score equal to or greater than 10 for the Conqueror class in the BrainHex questionnaire. The results from the feature selection can be seen in fig. 4.5. In the left column, we can

[1] *BestFirst* performs greedy hill climbing with backtracking; you can specify how many consecutive nonimproving nodes must be encountered before the system backtracks. [14]

[2] This algorithm works by searching for features that are highly correlated with the class, yet uncorrelated with each other

see the percentage of times, the given attribute was selected as ideal, for all ten folds. If an attribute was selected 20% of the time, it means that out of the ten folds performed, the attribute was selected two times, meaning that for a higher percentage of selections, we should have a better predictor.

```
number of folds (%)  attribute
          0(  0 %)     1 achiever_challenge
          0(  0 %)     2 survivor_challenge
          0(  0 %)     3 mastermind_challenge
          1( 10 %)     4 seeker_challenge
          0(  0 %)     5 daredevil_challenge
          0(  0 %)     6 conqueror_challenge
          6( 60 %)     7 path_choice_1_c
          0(  0 %)     8 path_choice_1_s
          9( 90 %)     9 path_choice_2_c
          0(  0 %)    10 path_choice_2_m
          0(  0 %)    11 path_choice_2_d
          9( 90 %)    12 path_choice_3_c
          0(  0 %)    13 path_choice_3_m
          0(  0 %)    14 path_choice_3_s
          0(  0 %)    15 path_choice_4_s
          0(  0 %)    16 path_choice_4_m
          1( 10 %)    17 path_choice_4_d
          8( 80 %)    18 path_choice_5_c
          0(  0 %)    19 path_choice_5_s
          0(  0 %)    20 path_choice_5_d
          0(  0 %)    21 sum_paths
          9( 90 %)    22 seeker_quest
          1( 10 %)    23 achiever_armor
          0(  0 %)    24 pots_collected
          0(  0 %)    25 coins_collected
          2( 20 %)    26 enemies_defeated
          0(  0 %)    27 seeker_secrets
```

Figure 4.5: Feature selection results for the Conqueror class dataset.

In this case, the attributes with any selection rates (selected at least once), are more or less inline with our expectations for the Conqueror class. Out of the six expected attributes for the conqueror class ("conqueror_challenge", "path_choice_1_c", "path_choice_2_c", "path_choice_3_c", "path_choice_4_c", "enemies_defeated"), only one ("conqueror_challenge") was not selected a single time, and five of them had a selection rate of at least 60%, with a mean of 82%. Of the five attributes with high selection rates (greater than or equal to 60%), all are paths, and the lowest is situated in the first intersection of the game, where the player is expected to explore more of the world around them, which might explain the lower result in comparison to the other paths.

Upon closer inspection of our training data we verified that all participants which were identified as Conquerors, in the BrainHex questionnaire, and did not complete the "conqueror_challenge", had very low participation rates in all challenges. This indicates they either didn't complete any challenge, or had at most one of the challenges completed. This didn't occur in our data for any other BrainHex class, so it could also serve as an indicator of the conqueror class. With this in mind, we decided to add a data-point exclusive to our conqueror dataset which merged all challenges' scores in a variable called "sum_challenges", that, as the name indicates, adds the scores of all challenges together.

4.3.2 Achiever

number of folds (%)	attribute
10(100 %)	1 achiever_challenge
0(0 %)	2 survivor_challenge
0(0 %)	3 mastermind_challenge
0(0 %)	4 seeker_challenge
0(0 %)	5 daredevil_challenge
0(0 %)	6 conqueror_challenge
0(0 %)	7 path_choice_1_c
0(0 %)	8 path_choice_1_s
0(0 %)	9 path_choice_2_c
0(0 %)	10 path_choice_2_m
0(0 %)	11 path_choice_2_d
0(0 %)	12 path_choice_3_c
0(0 %)	13 path_choice_3_m
0(0 %)	14 path_choice_3_s
0(0 %)	15 path_choice_4_s
0(0 %)	16 path_choice_4_m
0(0 %)	17 path_choice_4_d
0(0 %)	18 path_choice_5_c
0(0 %)	19 path_choice_5_s
0(0 %)	20 path_choice_5_d
1(10 %)	21 sum_paths
0(0 %)	22 seeker_quest
3(30 %)	23 achiever_armor
9(90 %)	24 pots_collected
1(10 %)	25 coins_collected
0(0 %)	26 enemies_defeated
6(60 %)	27 seeker_secrets

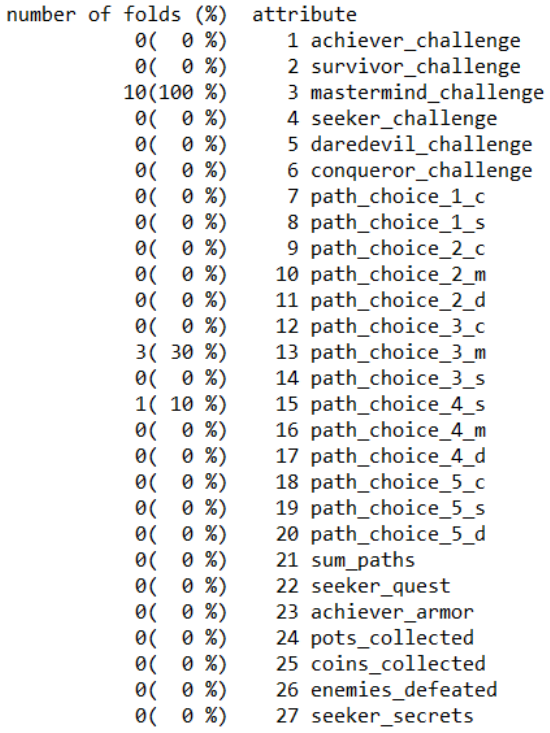
Figure 4.6: Feature selection results for the Achiever class dataset.

In this case, fig. 4.6, all attributes we might expect to see selected for the Achiever class ("achiever_challenge", "achiever_armor", "pots_collected", and "coins_collected"), have a selection rate greater than 0%, meaning they were identified. Out of the six identified attributes two were not part of our expectations: "sum_paths" and "seeker_secrets". The "sum_paths" attribute selection can be attributed to random noise as it was only selected once. On the other hand, the "seeker_secrets" might be showing up as a good attribute since some of the hidden locations were observable from the main path, but with their entrance hidden, making the player have to explore to find it. However, since the player was able to see the location, they might have coveted the rewards which were visible to them, encouraging the Achiever mindset of collecting everything [3].

4.3.3 Mastermind

In this case, fig. 4.7, out of all the expected attributes ("mastermind_challenge", "path_choice_2_m", "path_choice_3_m" and "path_choice_4_m"), only two were selected: "mastermind_challenge" and "path_choice_3_m". The other only selection made was "path_choice_4_s" (a survivor class type path), which with a selection

[3] This Achiever behavior is described in the official BrainHex website (https://onlyagame.typepad.com/brainhex/achiever.html), as "Your behaviour works towards the satisfaction of completing tasks and collections, and the intense reward of overcoming impossibly distant goals – about which you can become obsessive."



number of folds (%)	attribute
0(0 %)	1 achiever_challenge
0(0 %)	2 survivor_challenge
10(100 %)	3 mastermind_challenge
0(0 %)	4 seeker_challenge
0(0 %)	5 daredevil_challenge
0(0 %)	6 conqueror_challenge
0(0 %)	7 path_choice_1_c
0(0 %)	8 path_choice_1_s
0(0 %)	9 path_choice_2_c
0(0 %)	10 path_choice_2_m
0(0 %)	11 path_choice_2_d
0(0 %)	12 path_choice_3_c
3(30 %)	13 path_choice_3_m
0(0 %)	14 path_choice_3_s
1(10 %)	15 path_choice_4_s
0(0 %)	16 path_choice_4_m
0(0 %)	17 path_choice_4_d
0(0 %)	18 path_choice_5_c
0(0 %)	19 path_choice_5_s
0(0 %)	20 path_choice_5_d
0(0 %)	21 sum_paths
0(0 %)	22 seeker_quest
0(0 %)	23 achiever_armor
0(0 %)	24 pots_collected
0(0 %)	25 coins_collected
0(0 %)	26 enemies_defeated
0(0 %)	27 seeker_secrets

Figure 4.7: Feature selection results for the Mastermind class dataset.

rate of 10% we could attribute to random noise.

4.3.4 Survivor

In this case, fig. 4.8, out of the five expected attributes (“survivor_challenge”, “path_choice_1_s”, “path_choice_3_s”, “path_choice_4_s” and “path_choice_5_s”), one was not selected: “path_choice_1_s”, which like discussed in section 4.3.1, could be attributed to the fact that it is situated in the first intersection of the game, where the players are more likely to roam around exploring the world to experience the game, thus creating more noise. Out of the four selected attributes which were not expected, the “path_choice_1_c” is located in the first intersection, so it’s inclusion could be credited to random noise, and two of them have very low selection rates (10%).

4.3.5 Seeker

In this case, fig. 4.9, of the four expected attributes, three were selected at least 90% of the time (“seeker_challenge”, “seeker_quest”, “seeker_secrets”), while one was not selected a single time (“sum_paths”). Of the other 4 attributes selected, two were only selected once. This might means the “sum_paths” attribute is not a good indicator of the Seeker class.

```
number of folds (%)  attribute
           0(  0 %)     1 achiever_challenge
          10(100 %)     2 survivor_challenge
           1( 10 %)     3 mastermind_challenge
           0(  0 %)     4 seeker_challenge
           4( 40 %)     5 daredevil_challenge
           0(  0 %)     6 conqueror_challenge
           2( 20 %)     7 path_choice_1_c
           0(  0 %)     8 path_choice_1_s
           0(  0 %)     9 path_choice_2_c
           0(  0 %)    10 path_choice_2_m
           0(  0 %)    11 path_choice_2_d
           0(  0 %)    12 path_choice_3_c
           1( 10 %)    13 path_choice_3_m
           2( 20 %)    14 path_choice_3_s
           2( 20 %)    15 path_choice_4_s
           0(  0 %)    16 path_choice_4_m
           0(  0 %)    17 path_choice_4_d
           0(  0 %)    18 path_choice_5_c
           7( 70 %)    19 path_choice_5_s
           0(  0 %)    20 path_choice_5_d
           0(  0 %)    21 sum_paths
           0(  0 %)    22 seeker_quest
           0(  0 %)    23 achiever_armor
           0(  0 %)    24 pots_collected
           0(  0 %)    25 coins_collected
           0(  0 %)    26 enemies_defeated
           0(  0 %)    27 seeker_secrets
```

Figure 4.8: Feature selection results for the Survivor class dataset.

```
number of folds (%)  attribute
          10(100 %)     1 achiever_challenge
           4( 40 %)     2 survivor_challenge
           1( 10 %)     3 mastermind_challenge
          10(100 %)     4 seeker_challenge
           0(  0 %)     5 daredevil_challenge
           0(  0 %)     6 conqueror_challenge
           0(  0 %)     7 path_choice_1_c
           0(  0 %)     8 path_choice_1_s
           0(  0 %)     9 path_choice_2_c
           1( 10 %)    10 path_choice_2_m
           0(  0 %)    11 path_choice_2_d
           0(  0 %)    12 path_choice_3_c
           0(  0 %)    13 path_choice_3_m
           0(  0 %)    14 path_choice_3_s
           0(  0 %)    15 path_choice_4_s
           0(  0 %)    16 path_choice_4_m
           0(  0 %)    17 path_choice_4_d
           0(  0 %)    18 path_choice_5_c
           0(  0 %)    19 path_choice_5_s
           0(  0 %)    20 path_choice_5_d
           0(  0 %)    21 sum_paths
          10(100 %)    22 seeker_quest
           0(  0 %)    23 achiever_armor
           0(  0 %)    24 pots_collected
           1( 10 %)    25 coins_collected
           0(  0 %)    26 enemies_defeated
           9( 90 %)    27 seeker_secrets
```

Figure 4.9: Feature selection results for the Seeker class dataset.

4.3.6 Daredevil

```
number of folds (%)  attribute
           8( 80 %)    1 achiever_challenge
           0(  0 %)    2 survivor_challenge
           0(  0 %)    3 mastermind_challenge
           0(  0 %)    4 seeker_challenge
           0(  0 %)    5 daredevil_challenge
           0(  0 %)    6 conqueror_challenge
           0(  0 %)    7 path_choice_1_c
           0(  0 %)    8 path_choice_1_s
           0(  0 %)    9 path_choice_2_c
           0(  0 %)   10 path_choice_2_m
           0(  0 %)   11 path_choice_2_d
           0(  0 %)   12 path_choice_3_c
           0(  0 %)   13 path_choice_3_m
           0(  0 %)   14 path_choice_3_s
           0(  0 %)   15 path_choice_4_s
           0(  0 %)   16 path_choice_4_m
           1( 10 %)   17 path_choice_4_d
           0(  0 %)   18 path_choice_5_c
           0(  0 %)   19 path_choice_5_s
           0(  0 %)   20 path_choice_5_d
           0(  0 %)   21 sum_paths
           0(  0 %)   22 seeker_quest
           0(  0 %)   23 achiever_armor
           0(  0 %)   24 pots_collected
           0(  0 %)   25 coins_collected
           0(  0 %)   26 enemies_defeated
           1( 10 %)   27 seeker_secrets
```

Figure 4.10: Feature selection results for the Daredevil class dataset.

In this case, fig. 4.10, of all expected attributes ("daredevil_challenge", "path_choice_2_d", "path_choice_4_d" and "path_choice_5_d"), only one was selected, and it only happened in one of the ten folds. Of the other two attributes selected, only "achiever_challenge" had a high selection rate (80%), while the other ("seeker_secrets"), only had a 10% selection rate. This might mean that our attributes do not have a high correlation with the class on their own. This, however, does not mean that, as a whole, this attributes cannot be good indicators of the class. To answer this question we will need the results of the machine learning algorithms we will employ next.

4.4 Model Training Results

With all datasets properly formatted and the information from the feature selection to help guide our choices, we started training our models. The first algorithm we tried with the datasets was the RepTree [16] algorithm, which uses information gain/variance reduction to build a decision tree. The second algorithm tested was the RandomTree algorithm, which considers K randomly chosen attributes at each node to construct a decision tree. The third algorithm tested was the RandomForest [26], which makes a forest of random trees. The fourth algorithm tested was the Naive Bayes classifier [27]. The fifth

and last algorithm tested was KMeans clustering [28], which clusters the data using the K means algorithm discussed in section 2.3.1.B. The results obtained from running these algorithm using 10-fold cross validation are presented across different levels of dataset stages, and obtained after performing hyperparameter tunning. The first stage of the dataset consists in the default dataset with all the variables previously discussed in section 4.2.1 and section 4.3.1, giving us modest results across the three algorithms. The second stage comes from performing feature selection, since it showed to be promising in section 4.3, discretizing the data for Naive Bayes [27] with the help of WEKA's *Discretize* filter [4], and running the algorithms again in the new dataset. The evaluation for the KMeans algorithm was done via the classes to cluster feature, where WEKA first ignores the class and builds the clusters. It then allocates classes to the clusters during the test phase, depending on the majority value of the class attribute inside each cluster. Finally, depending on this assignment, it computes the classification error which we subtract from 100% to obtain the accuracy, this being the value displayed in the tables below.

4.4.1 Conqueror

The results obtained from running these algorithm using 10-fold cross validation across different levels of dataset stages for the Conqueror dataset can be seen in table 4.4. As we can observe, the feature selection made to the original dataset produced better results in four out of five algorithms. This might indicate that, there might be some relevant information in the dataset that is being removed while moving from *Stage 1* to *Stage 2* (performing feature selection), that is being captured by the RandomForest algorithm, but that with such a small dataset might seem like random noise to other more simplistic algorithms. Although we are unable to verify the following statement since observing the behavior of all decision trees in the forest is not feasible, we think this might occur because the RandomForest algorithm makes use of several decision trees to weight the classification, while the other two tree methods only make use of one tree, giving more opportunities to pick up more subtle relations to between the attributes measured and the class. A possible example would be the a tree that uses an attribute which, in conjunction with another, might reveal a subtle behavior of the player that is manifested by some of the participants identified as the given class. Since this attributes are not highly and directly correlated to the class, and not all participants show the given behavior, they are not being selected in more simple tree algorithms, with such a low sample rate.

4.4.2 Achiever

The results obtained from running these algorithm using 10-fold cross validation across different levels of dataset stages for the Achiever dataset can be seen in table 4.5. As we can observe, the feature

[4] This instance filter works by discretizing a range of numeric attributes in the dataset into nominal attributes. Discretization is by simple binning. Skips the class attribute if set. [14]

Stages	RepTree	RandomTree	RandomForest	NaiveBayes	KMeans
1	62.50%	83.33%	83.33%	75.00%	50.00%
2	75.00%	87.50%	79.16%	79.16%	79.16%

Table 4.4: Conqueror dataset model training results for the five machine learning algorithms used, in both *Stage 1* and *Stage 2* of the datasets. The numbers presented are the accuracy of each model as given by performing 10-fold cross validation.

selection made to the original dataset produced better results for all five algorithms, consolidating the idea that our choice for the metrics used to measure the BrainHex Achiever class was appropriate.

Stages	RepTree	RandomTree	RandomForest	NaiveBayes	KMeans
1	66.66%	75.00%	79.16%	75.00%	58.34%
2	70.83%	79.16%	83.33	79.16%	70.83%

Table 4.5: Achiever dataset model training results for the five machine learning algorithms used, in both *Stage 1* and *Stage 2* of the datasets. The numbers presented are the accuracy of each model as given by performing 10-fold cross validation.

4.4.3 Mastermind

The results obtained from running these algorithm using 10-fold cross validation across different levels of dataset stages for the Mastermind dataset can be seen in table 4.6. As we can observe, the feature selection made to the original dataset produced better results for three of the five algorithms, with one remaining the same, and another going down. Much like in the Conqueror dataset, we think that the RandomForest algorithm might be picking up some relation between the attributes removed with the feature selection and the class, that may be too subtle for the other algorithms to pick up with such a small dataset as ours.

Stages	RepTree	RandomTree	RandomForest	NaiveBayes	KMeans
1	75.00%	75.00%	79.16%	70.83%	66.66%
2	83.33%	79.16%	75.00%	79.16%	66.66%

Table 4.6: Mastermind dataset model training results for the five machine learning algorithms used, in both *Stage 1* and *Stage 2* of the datasets. The numbers presented are the accuracy of each model as given by performing 10-fold cross validation.

4.4.4 Survivor

The results obtained from running these algorithm using 10-fold cross validation across different levels of dataset stages for the Survivor dataset can be seen in table 4.7. As we can observe, the feature selection made to the original dataset produced better results for three of the five algorithms, with the

other two remaining the same, consolidating the idea that our choice for the metrics used to measure the BrainHex Survivor class was appropriate.

Stages	RepTree	RandomTree	RandomForest	NaiveBayes	KMeans
1	87.50%	79.16%	79.16%	87.50%	62.50%
2	87.50%	79.16%	87.50%	91.66%	70.83%

Table 4.7: Survivor dataset model training results for the five machine learning algorithms used, in both *Stage 1* and *Stage 2* of the datasets. The numbers presented are the accuracy of each model as given by performing 10-fold cross validation.

4.4.5 Seeker

The results obtained from running these algorithm using 10-fold cross validation across different levels of dataset stages for the Seeker dataset can be seen in table 4.8. As we can observe, the feature selection made to the original dataset produced better results for all of the five algorithms that didn't already have a baseline of 100%, consolidating the idea that our choice for the metrics used to measure the BrainHex Seeker class was appropriate. this results are because two indicators (seekrcahllenge,

Stages	RepTree	RandomTree	RandomForest	NaiveBayes	KMeans
1	100%	95.83%	100%	91.66%	66.66%
2	100%	100%	100%	95.83%	91.67%

Table 4.8: Seeker dataset model training results for the five machine learning algorithms used, in both *Stage 1* and *Stage 2* of the datasets. The numbers presented are the accuracy of each model as given by performing 10-fold cross validation.

and seekerquest) were very good. with more data it might not happen

4.4.6 Daredevil

The results obtained from running these algorithm using 10-fold cross validation across different levels of dataset stages for the Daredevil dataset can be seen in table 4.9. As we can observe, the feature selection made to the original dataset produced better results for all of the five algorithms. We can observe from this results, in comparison to the other datasets, that while our choice for the metrics used to measure the BrainHex Daredevil class seems relatively appropriate, considering the increase in performance of the models after performing feature selection, there probably are more metrics we should have considered for the Daredevil class. One of the metrics we originally wanted to consider but ended up removing how fast the players were moving around throughout the game, especially in the Daredevil section. We ended up removing this metric, since several players reported taking small

breaks during the playtesting given how long the experiment took, which could have severely skewed the results.

Stages	RepTree	RandomTree	RandomForest	NaiveBayes	KMeans
1	58.33%	58.33%	66.66%	62.5%	62.5%
2	75.00%	75.00%	70.83%	75.00%	66.66%

Table 4.9: Daredevil dataset model training results for the five machine learning algorithms used, in both *Stage 1* and *Stage 2* of the datasets. The numbers presented are the accuracy of each model as given by performing 10-fold cross validation.

4.5 Model Validation Results

The best models were recorded for each algorithm of each dataset, and used in the final validation performed with the remaining 30% of the data (testing/validation dataset) as discussed in section 4.2.2. These results can be seen in table 4.10. As we can observe in the table, five out of the six classes had very high levels of accuracy, greater than or equal to 83.33%, which means at least five out of the six instances in the validation dataset were correctly identified. On the other hand, the Daredevil class had very low levels of accuracy, which like explained in section 4.4.6, might mean our choice of metrics and challenges for the class were not appropriate. Since WEKA doesn't support test set evaluations for KMeans clustering we are unable to provide validation results for this algorithm. We tried doing it by hand, but with such a large number of attributes it would not be feasible to do it for all six participants for all datasets. However, since the clustering creation is unsupervised, like discussed in section 4.4, the results obtained in section 4.4 still hold some value.

Datasets	RepTree	RandomTree	RandomForest	NaiveBayes	KMeans
Conqueror	100%	100%	83.33%	83.33%	–%
Achiever	83.33%	83.33%	83.33%	100%	–%
Mastermind	83.33%	83.33%	83.33%	100%	–%
Survivor	83.33%	83.33%	83.33%	83.33%	–%
Seeker	100%	100%	100%	83.33%	–%
Daredevil	50%	50%	66.66%	66.66%	–%

Table 4.10: Model validation results for the six different datasets using the best performing models seen in section 4.4 for each algorithm of each dataset.

Regarding the classes with high success rates, Seeker performed especially well both under training and validation. Upon closer inspection we could observe that the attributes "seeker_challenge" and "seeker_quest" discussed in section 4.3, proved to be excellent indicators of Seeker behavior, as both had clear cut-off values, which perfectly split the dataset, identifying the correct class. Although our dataset had a small number of samples (n=30, with n=24 for the testing set and n=6 for validation), and

these cut-offs might not have worked perfectly with a higher number of samples, we can conclude these attributes are very good indicators for identifying the class in question. As for the other three classes with high scores, we can see that they all obtained around the same accuracy in both the training and validation phases, taking into account the low number of samples in our dataset, which restricts the number of possible steps of accuracy that can be obtained.

```
path_choice_2_c < 0.9 : 0 (8/0)
path_choice_2_c >= 0.9
|   seeker_quest < 0.82
|   |   path_choice_3_c < 0.5 : 0 (1/0)
|   |   path_choice_3_c >= 0.5
|   |   |   path_choice_1_c < 0.6 : 0 (1/0)
|   |   |   path_choice_1_c >= 0.6
|   |   |   |   enemies_defeated < 0.4
|   |   |   |   |   enemies_defeated < 0.33 : 1 (2/0)
|   |   |   |   |   enemies_defeated >= 0.33 : 0 (1/0)
|   |   |   |   enemies_defeated >= 0.4 : 1 (9/0)
|   seeker_quest >= 0.82 : 0 (2/0)
```

Figure 4.11: The Random Tree which provided the best results in 10-fold cross validation for the Conqueror dataset.

An example of a RandomTree model created during the training process can be seen in fig. 4.11, which pictures the tree model used in the RandomTree classification with 100% accuracy for the Conqueror dataset in table 4.10. This tree, much like the feature selection performed in section 4.3, includes mostly attributes which were desgined specifically with the Conqueror class in mind, with the one exception being “seeker_quest”. Although we require more data to come up with a solid conclusion we think that, this attribute (“seeker_quest”) might not be an anomaly, but an indication that players who identify as Conquerors might not like that type of content. Other anomalies like this one might have been identifiable in other instances if we had more data, raising questions on the existence of some level of interconnection between different BrainHex classes, or indicating some overlap in the design decisions made for the given challenge. As for the other three classes with high scores, we can see that they all obtained around the same accuracy in both the training and validation phases, taking into account the low number of samples in our dataset, which restricts the number of possible steps of accuracy that can be obtained.

After finalizing the dataset validation we decided to create a list of the top indicators, and a possible explanation for some of them, for each BrainHex class based on the final Decision Tree models and the feature selection.

Conqueror:

- The four paths that came from the main intersections, which excludes the first path as it was in the exploration section of the game. ´
- The number of enemies defeated.

- The lack of engagement with the Seeker quest.

Survivor:

- The engagement with the Survivor challenge.
- The engagement with the last Survivor path. The first three Survivor paths after the exploration zone where neither good nor bad indicators.
- The non engagement with one of the mastermind paths.

Seeker:

- The engagement with the Seeker challenge.
- The engagement with the Seeker quest.
- The number of hidden zones visited.

Achiever:

- The engagement with the Achiever challenge.
- The number of pots and coins collected.
- The engagement with the Achiever armor collection challenge.
- The number of hidden zones reached. This metric is related to the hidden zones which where visible from the normal paths and the player had to find the entrance to. They had rewards that the player could see (such as chests, coins, or pots, among others), which incentivized the players to go collect them.

Daredevil:

- The Daredevil's paths were mediocre indicators.
- The engagement with the Daredevil challenge was a mediocre indicator.

Mastermind:

- The engagement with the Mastermind challenge.
- The second and third Mastermind paths after the exploration zone. The other paths were not so good indicators.

4.6 Discussion

In this chapter we presented and discussed our results from the final experiment. We concluded that, for all BrainHex classes besides Daredevil, we were able to correctly identify the participants BrainHex type. This result corroborates our hypothesis that it is possible to extract the players' self reported preferences from their in-game behavior. This allowed for the creation of several machine learning models, using different algorithms, which could identify the players' BrainHex class, from data collected of their in-game behavior, with a high degree of accuracy.

5

Conclusion

Contents

5.1 Discussion

In order to widen the range of target players for games, developers can employ the use PCG to generate content tailored to the player. Since the generation of tailored content requires the acquisition of information regarding the player's preferences, developers can ask the players' to fill out validated player models questionnaires to obtain it. This, however, comes with the problem that most players won't be willing to spend time filling out a form before they can start engaging with the game.

In order to solve this problem, we developed a proof of concept system which is able to identify the players' preferences. We based our work on the BrainHex player model, removing the Social element since we worked on an offline single-player game. With the remaining six BrainHex classes, we designed challenges tailored to each one of them and created an entire game around it. From this game, we collected data pertaining to the players' choices and actions taken while playing it, which was then treated to properly fit our purposes. This treatment included the creation of six different datasets, all copied from the original one, but differing in the target class, with each one corresponding to one of the six BrainHex classes we studied.

This means that, in a game that uses PCG to generate content tailored to the player, in the first stage of a game, the player would be prompted to express his preferences in a section specifically designed for this purpose. The game would thereafter be procedurally generated using the preferences extracted from the player's behavior.

We selected several machine learning algorithms, REP Tree, Random Tree, Random Forest, Naive Bayes, and K-Means, which were employed with the help of our selected machine learning software, WEKA. We started by training models with all algorithms for each one of the six datasets. Afterwards, these models were validated with the 30% testing dataset.

The results were positive, with high accuracy, for five of the six datasets, with the Daredevil class dataset performing significantly worse than the other five. The K-Means algorithm proved to be the worse performer in almost all cases, raising into question its acceptability for use with the type of data collected since it requires a more spatial structure of the data that might not have been present in our datasets. It could also be that the clusters were not uniform or spherical, which might require the use of other clustering algorithms.

In our opinion, our results corroborate our hypothesis, that players will exhibit their gaming preferences while playing a game and that these preferences can be extracted from their behavior. This requires careful consideration during the design process of the game, to make sure the preferences which we want to measure are well represented inside the game. It is also important to make sure the metrics recorded are relevant to the type of preference we are measuring.

Finally, we expect this work can shed light on the process required to acquire relevant information for the generation of content using PCG, without the need to employ a questionnaire.

5.2 Future Work

For future research, we suggest the replication of this study, first and foremost, with a focus on obtaining more participants. We also had other ideas on how to improve or expand our work such as:

- Experiment with the methods described here using different player type models, such as GMP, Marczewski's Player and User Types Hexad or Bartle Taxonomy of Player Types, or even with personality models, such as the FFM or MBTI.
- Exploring the potential relation between different BrainHex classes. We saw this happen in one instance, where the non-interaction with a challenge from the Seeker BrainHex class proved to be an indicator for the player belonging to the Conqueror class.
- Better representation of different classes throughout the dataset. Some classes representation was not perfect, and with a small dataset like ours, one outlier can more easily skew the results.
- The creation of the PCG system for the generation of tailored content using the models created. This is the logical next step in the process of providing a truly tailored experience to the player without the need to fill any questionnaires before playing.

Furthermore, we advise extra care with the design of challenges and the metrics recorded from them, since they will have the greatest impact on the results obtained from the models. In this line of thought, we also advise careful consideration to the freedom given to the player, and to be mindful of player curiosity and exploration during the initial section of the game.

Bibliography

[1] R. Dias and C. Martinho, “inflow: Adapting gameplay to player personality,” 2010.

[2] B. Odierna and I. Silveira, *PLAYER GAME DATA MINING FOR PLAYER CLASSIFICATION*, 05 2019, pp. 52–61.

[3] L. F. Bicalho, A. Baffa, and B. Feijó, “A game analytics model to identify player profiles in single-player games,” in *2019 18th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, 2019, pp. 11–20.

[4] E. J. Hastings, R. K. Guha, and K. O. Stanley, “Evolving content in the galactic arms race video game,” in *2009 IEEE Symposium on Computational Intelligence and Games*, 2009, pp. 241–248.

[5] N. Shaker, G. Yannakakis, and J. Togelius, “Towards automatic personalized content generation for platform games,” 12 2010.

[6] J. Togelius, R. De Nardi, and S. M. Lucas, “Towards automatic personalised content creation for racing games,” in *2007 IEEE Symposium on Computational Intelligence and Games*, 2007, pp. 252–259.

[7] I. B. Myers and P. B. Myers, *Gifts differing: understanding personality type*. Consulting Psychologists Press, 1995.

[8] R. R. McCrae and O. P. John, “An introduction to the five-factor model and its applications,” *Journal of Personality*, vol. 60, no. 2, p. 175–215, Jun 1992. [Online]. Available: http://dx.doi.org/10.1111/j.1467-6494.1992.tb00970.x

[9] R. Bartle, “Hearts, clubs, diamonds, spades: Players who suit muds,” 1996.

[10] N. Yee, “The gamer motivation profile: What we learned from 250,000 gamers,” 2016.

[11] A. Marczewski, *User Types HEXAD*, 2015.

[12] L. E. Nacke, C. Bateman, and R. L. Mandryk, “Brainhex: A neurobiological gamer typology survey,” *Entertainment Computing*, vol. 5, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1875952113000086

[13] Intenational Hobo. (2008) BrainHex. [Online]. Available: https://blog.brainhex.com/

[14] Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2017)., *Data mining: Practical machine learning tools and techniques. Amsterdam: Morgan Kaufmann.*

[15] J. R. Quinlan, “Induction of decision trees,” *MACH. LEARN*, vol. 1, pp. 81–106, 1986.

[16] ——, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[17] S. Kotsiantis, “Supervised machine learning: A review of classification techniques.” *Informatica (Slovenia)*, vol. 31, pp. 249–268, 01 2007.

[18] T. K. Ho, “Random decision forests,” in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, 1995, pp. 278–282 vol.1.

[19] T. Mahlmann, A. Drachen, J. Togelius, A. Canossa, and G. Yannakakis, “Predicting player behavior in tomb raider: Underworld (pre-print),” 09 2010, pp. 178 – 185.

[20] A. Drachen, R. Sifa, C. Bauckhage, and C. Thurau, “Guns, swords and data: Clustering of player behavior in computer games in the wild (pre-print),” 09 2012.

[21] I. Capelo and C. Martinho, “Glass: Adapting game content to player affective state and personality,” *Master Thesis, Instituto Superior Tecnico*, 2013.

[22] B. Almeida and C. Martinho, “Game content placement through progression modeling,” *Master Thesis, Instituto Superior Tecnico*, 2018.

[23] W. W. IJsselsteijn, de Yaw Yvonne Kort, and K. Poels, “The game experience questionnaire,” 2013.

[24] Microsoft Corporation, “Microsoft excel.” [Online]. Available: https://office.microsoft.com/excel

[25] M. A. Hall, “Correlation-based feature subset selection for machine learning,” Ph.D. dissertation, University of Waikato, Hamilton, New Zealand, 1998.

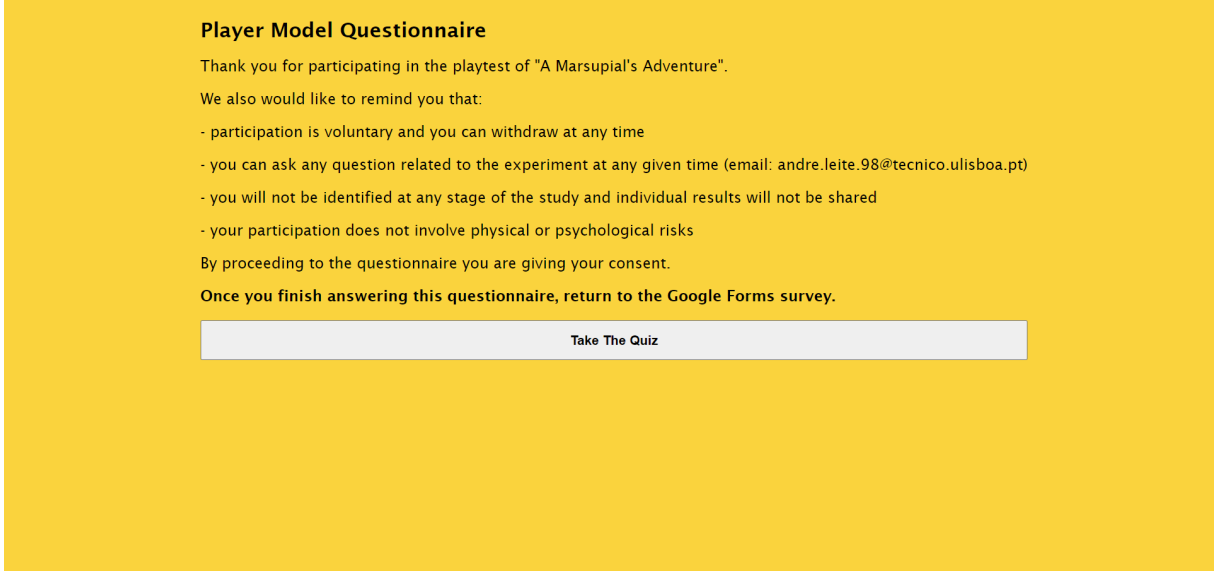
[26] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[27] G. H. John and P. Langley, “Estimating continuous distributions in bayesian classifiers,” in *Eleventh Conference on Uncertainty in Artificial Intelligence*. San Mateo: Morgan Kaufmann, 1995, pp. 338–345.

[28] D. Arthur and S. Vassilvitskii, “k-means++: the advantages of carefull seeding,” in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007, pp. 1027–1035.

A

BrainHex Questionnaire



Player Model Questionnaire

Thank you for participating in the playtest of "A Marsupial's Adventure".

We also would like to remind you that:

- participation is voluntary and you can withdraw at any time
- you can ask any question related to the experiment at any given time (email: andre.leite.98@tecnico.ulisboa.pt)
- you will not be identified at any stage of the study and individual results will not be shared
- your participation does not involve physical or psychological risks

By proceeding to the questionnaire you are giving your consent.

Once you finish answering this questionnaire, return to the Google Forms survey.

Take The Quiz

Player Model Questions

- 1 : Exploring to see what you can find. *
  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!
- 2 : Frantically escaping from a terrifying foe. *
  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!
- 3 : Working out how to crack a challenging puzzle. *
  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!
- 4 : The struggle to defeat a difficult boss. *
  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!
- 5 : Playing in a group, online or in the same room. *
  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!
- 6 : Responding quickly to an exciting situation. *
  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!
- 7 : Picking up every single collectible in an area. *
  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!
- 8 : Looking around just to enjoy the scenery. *
  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

- 9 : Being in control at high speed. *

  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

- 10 : Devising a promising strategy when deciding what to try next. *

  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

- 11 : Feeling relief when you escape to a safe area. *

  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

- 12 : Taking on a strong opponent when playing against a human player in a versus match. *

  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

- 13 : Talking with other players, online or in the same room. *

  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

- 14 : Finding what you need to complete a collection. *

  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

- 15 : Hanging from a high ledge. *

  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

- 16 : Wondering what's behind a locked door. *

  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

- 17 : Feeling scared, terrified or disturbed. *

  - I love it!
  - I like it.
  - It's okay.
  - I dislike it.
  - I hate it!

• 18 : Working out what to do on your own. *

- I love it!
- I like it.
- It's okay.
- I dislike it.
- I hate it!

• 19 : Completing a punishing challenge after failing many times. *

- I love it!
- I like it.
- It's okay.
- I dislike it.
- I hate it!

• 20 : Co-operating with strangers. *

- I love it!
- I like it.
- It's okay.
- I dislike it.
- I hate it!

• 21 : Getting 100% (completing everything in a game). *

- I love it!
- I like it.
- It's okay.
- I dislike it.
- I hate it!

• Please list the seven statements in order of preference, choosing each number only once: (1-worst, 7-best)

-- A moment of jaw-dropping wonder or beauty. *
-- An experience of primeval terror that blows your mind. *
-- A moment of breathtaking speed or vertigo. *
-- The moment when the solution to a difficult puzzle clicks in your mind. *
-- A moment of hard-fought victory. *
-- A moment when you feel an intense sense of unity with another player. *
-- A moment of completeness that you have strived for. *

• Now it's time to see your results...

Submit Quiz

Please write down this UID

and enter it inside the game:

6161e004d7819

Download the game!

B

Manipulation Check Questionnaire

Challenge Validation Form

Thank you for your participation in this study. This study is part of a Master's dissertation at Instituto Superior Técnico that aims to test some challenges developed for the game A Marsupial's Adventure.

The questionnaire will start with some demographic questions, followed by a selection of 6 videos (each around 1:00min - 1:30min) showcasing different challenges. You will be asked to state your agreement with sentences written underneath each video. You are free to rewatch each video several times and can change any answer at any time by scrolling up or down to the specific question.

Answering this will take, approximately, 20-25 minutes.

We also would like to remind you that:
- participation is voluntary and you can withdraw at any time;
- you can ask any question related to the experiment at any given time (email: andre.leite.98@tecnico.ulisboa.pt);
- you will not be identified at any stage of the study and individual results will not be shared;
- your participation does not involve physical or psychological risks.

By proceeding to the questionnaire you are giving your consent.

In advance, I want to thank you for your time.

* Required

Characterization

This section asks about you and your gaming habits.

1. Age *

2. Sex *

Mark only one oval.

- ◯ Male
- ◯ Female
- ◯ Other
- ◯ Prefer not to say

3. How often do you play video games *

Mark only one oval.

- ◯ I don't play video games
- ◯ I play video games occasionally when the opportunity presents itself.
- ◯ I make some time in my schedule to play video games.

A Marsupial's Adventure

A Marsupial's Adventure is a 2D Top-Down adventure game where you play as a koala.

This sections has 1 questions.

4. How familiar are you with games in which the protagonist combats a large number of enemies by shooting at them while dodging their fire, like Enter the Gungeon, Binding of Issac, Cuphead, Nuclear Throne? *

Mark only one oval.

- ◯ I am not familiar with these games and/or have no formed opinion on them.
- ◯ I played/watched others play them enough to understand I do not appreciate them.
- ◯ I enjoy these games and have played/watched others play them multiple times.

Challenges

Please watch each of the 6 videos (each around 50sec - 1:30min) and fill the table beneath.
There are links for a fullscreen version (preferred) of the videos in each challenge, however, you can still watch them directly here (not preferred).
Please be aware that the game is still in development and some assets are temporary.

This sections has 6 tables of questions.

Challenge A: https://youtu.be/IVHTWJKYtIU

http://youtube.com/watch?v=IVHTWJKYtIU

5. Challenge A - Rate each statement by how much you feel it relates to the challenge depicted in the video. *

Mark only one oval per row.

	Strongly Disagree	Disagree	Slightly Disagree	Neutral	Slightly Agree	Agree	Strongly Agree
You like negotiating dizzying platforms or rushing around at high speed while you are still in control.							
You like defeating impossibly difficult foes, struggling until you eventually achieve victory.							
You like solving puzzles and devising strategies.							
You like finding strange and wonderful things, or finding familiar things.							
You like escaping from hideous and scary threats, pulse-							

pounding risks.							
You like collecting anything you can collect, and doing everything you possibly can.							

C

Final Experiment Questionnaire

A Marsupial's Adventure Testing Questionnaire

Thank you for participating in the playtest of "A Marsupial's Adventure". This game was developed as part of a Master's dissertation at Instituto Superior Técnico.

First, you will have to answer some demographic and game-habits related questions. Next, you will proceed to answer a player model questionnaire, and then you will be tasked with playing the game "A Marsupial's Adventure". Finally, you will be asked to rate your experience.
More specific information will be provided to you throughout the experiment.

The whole activity will take, approximately, 40 minutes to 1 hour and 15 minutes, depending on how you play the game.

We also would like to remind you that:
- participation is voluntary and you can withdraw at any time;
- you can ask any question related to the experiment at any given time (email: andre.leite.98@tecnico.ulisboa.pt);
- you will not be identified at any stage of the study and individual results will not be shared;
- your participation does not involve physical or psychological risks.
By proceeding to the questionnaire you are giving your consent.

In advance, I want to thank you for your time.

* Required

Characterization

This section asks about you and your gaming habits.

1. Age *

2. Gender *

Mark only one oval.

- ◯ Male
- ◯ Female
- ◯ Prefer not to say
- ◯ Other: _____

3. How often do you play video games *

Mark only one oval.

- ◯ I don't play video games
- ◯ I play video games occasionally when the opportunity presents itself.
- ◯ I make some time in my schedule to play video games.

4. How familiar are you with games in which the protagonist combats a large number of enemies by shooting at them while dodging their fire, like Enter the Gungeon, Binding of Issac, Cuphead, Nuclear Throne? *

Mark only one oval.

- ◯ I am not familiar with these games and/or have no formed opinion on them.
- ◯ I played/watched others play them enough to understand I do not appreciate them.
- ◯ I enjoy these games and have played/watched others play them multiple times.

Player Model Questionnaire

In order to make sure we gather different types of players, we ask you to answer this player model questionnaire before playing the game.

At the end of the questionnaire, you will be given a UID (Unique Identifier).
There are two things you should do with said UID:
- Please write it down in the field below.
- Please save it somewhere so you can input it inside the game when it starts.

http://web.ist.utl.pt/ist186383

When you finish the questionnaire you will be given a link to download the game.

5. Please insert your UID here: *

Playing the game

Please download the game form the link given at the end of the player model questionnaire.

If for some reason you forgot to download the game from there, and already closed it, you can do so using the following link: https://drive.google.com/drive/folders/1mH9bDqsUPHfvKehBIoS4ZY_GEE_5YAxZ?usp=sharing

After downloading the game, please extract the contents form the zip and start it by opening the "Thesis.exe" file.

After playing the game, please come back to this questionnaire and answer the next section.

Game Experience Questionnaire

Please answer the following question regarding your experience playing the game.

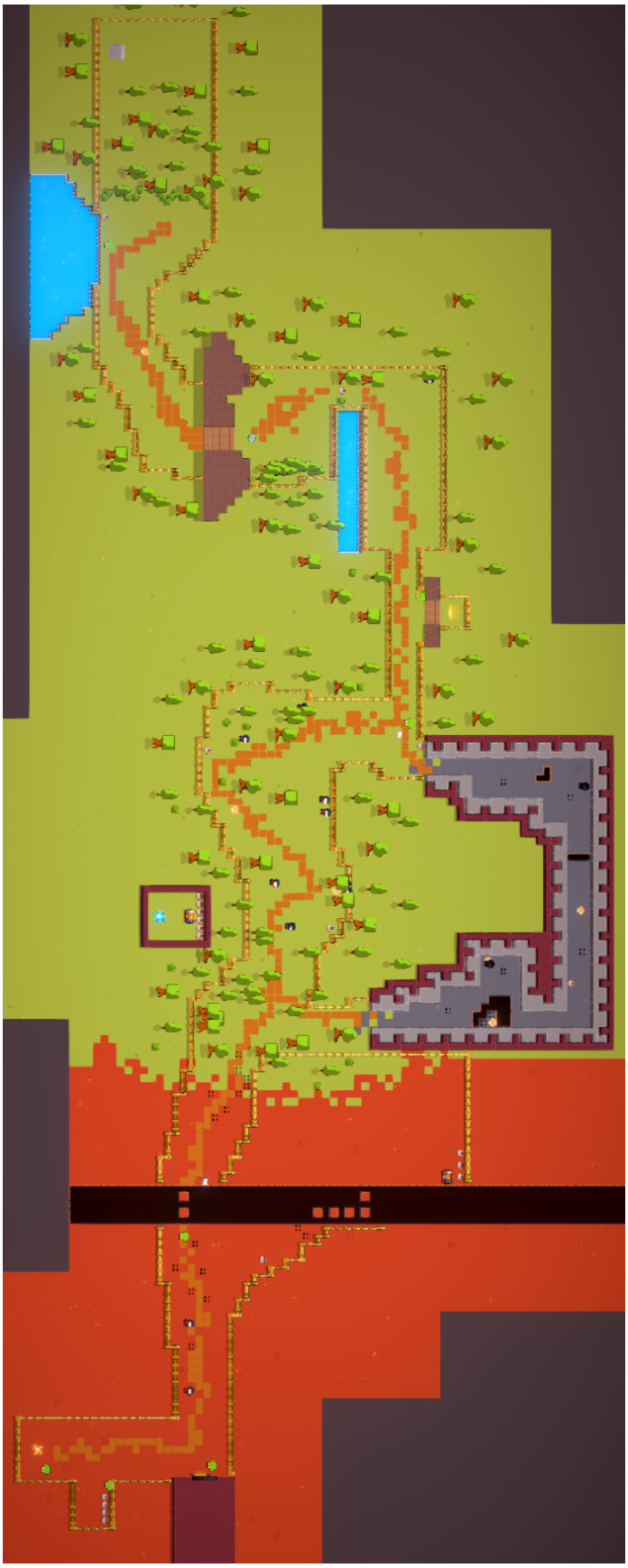
6. Please indicate how you felt while playing the game for each of the items, on the following scale: *

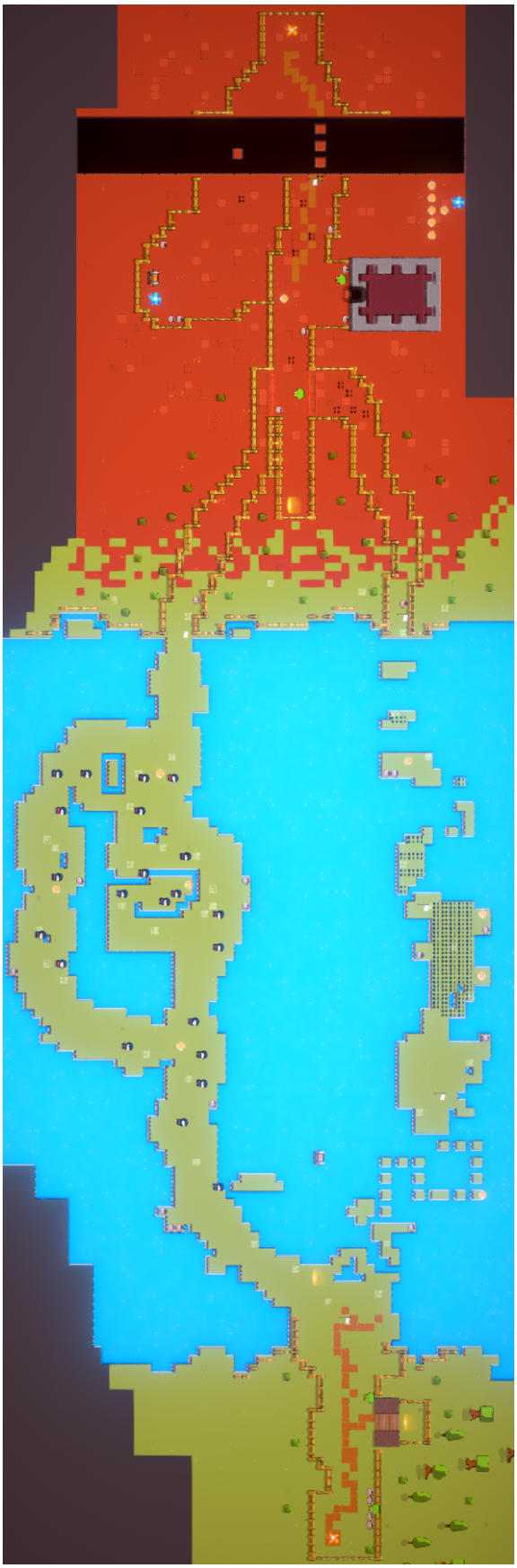
Mark only one oval per row.

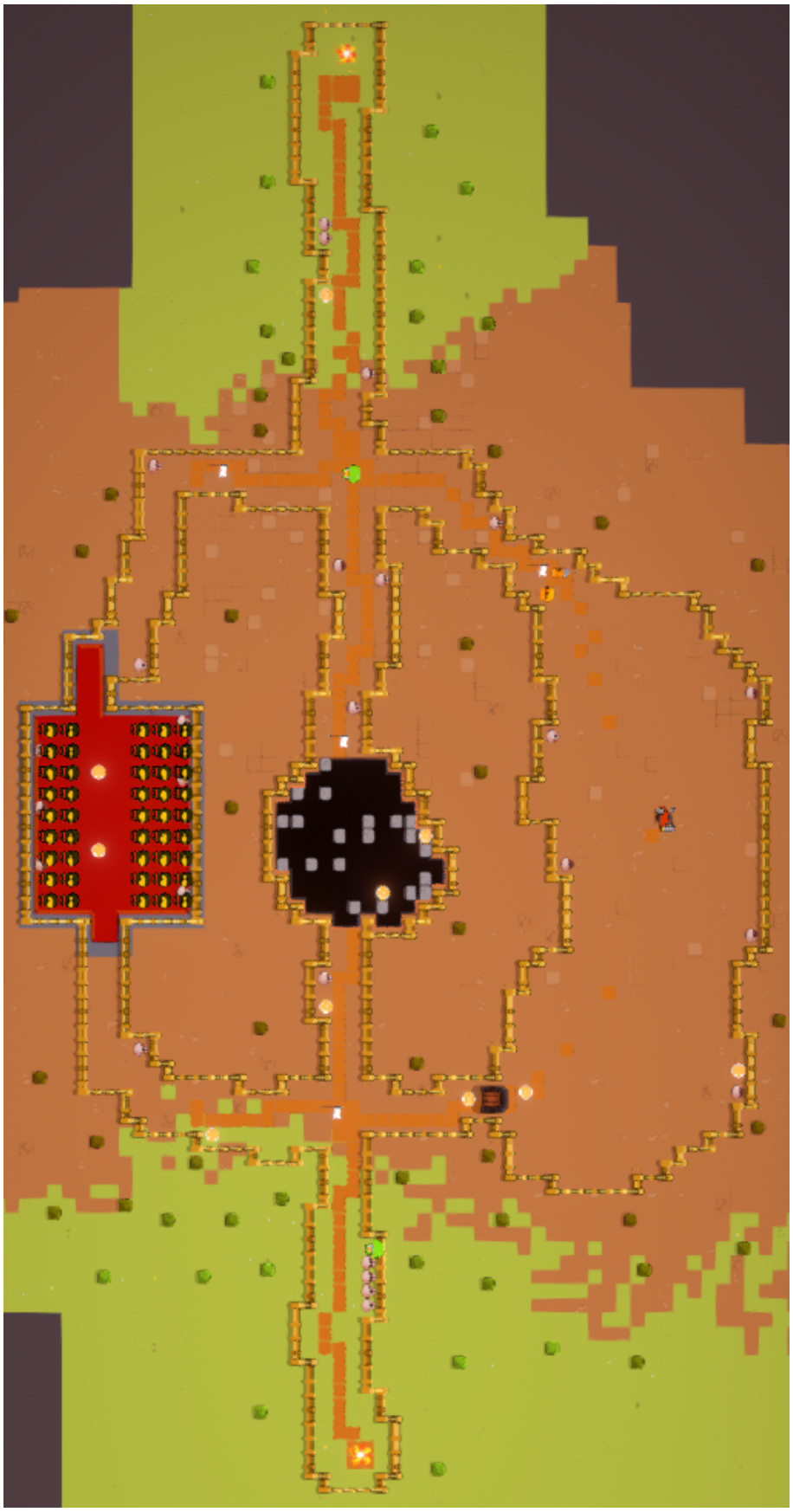
	not at all	slightly	moderatly	fairly	extremely
I was interested in the game's story					
I felt successful					
I felt bored					
I found it impressive					
I forgot everything around me					
I felt frustrated					
I found it tiresome					
I felt irritable					
I felt skilful					
I felt completely absorbed					
I felt content					
I felt challenged					
I had to put a lot of effort into it					
I felt good					

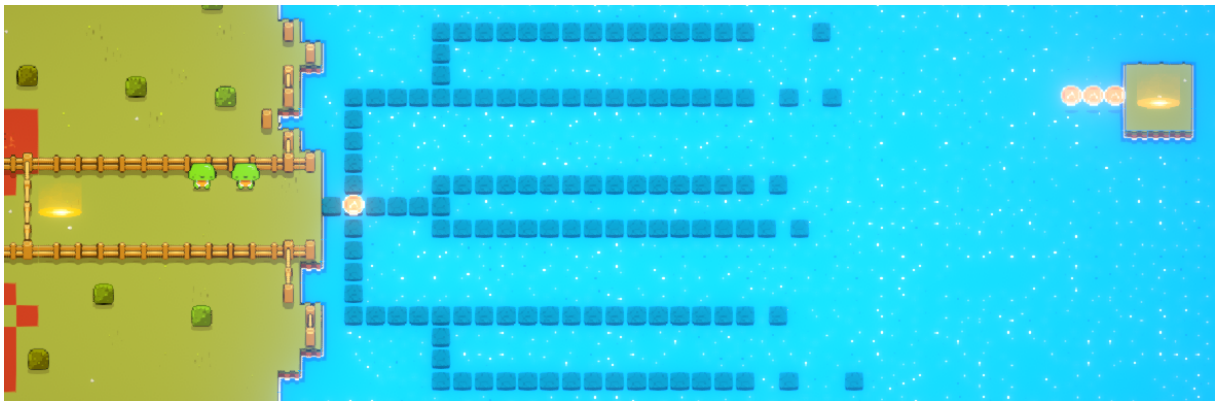
D

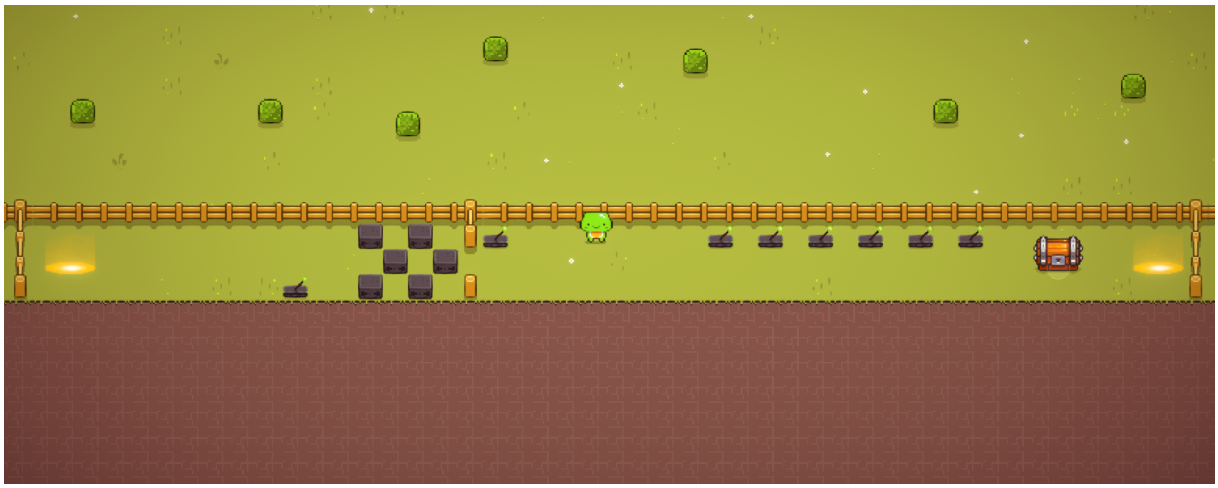
Map of the game

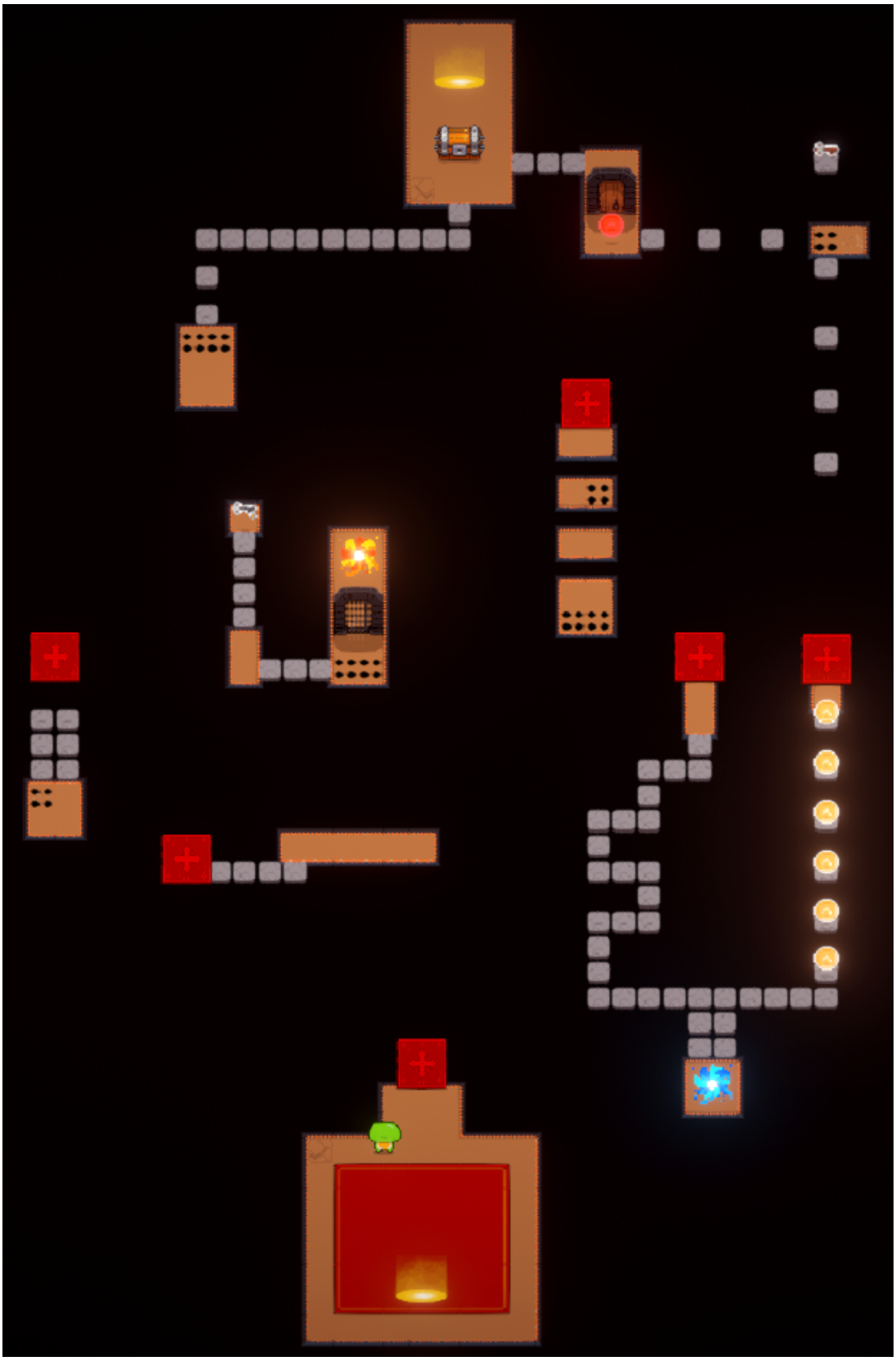












E

Example of a part of a raw log-data file.

Listing E.1: Log Data File Example

```
74.96053 : Checkpoint : 20 : Flag2
79.50283 : Inventory : Pick : Pot : 1
82.2854 : EnemyDeath : NinjaPatrol
84.54643 : EnemyDeath : NinjaPatrol
86.76768 : Inventory : Pick : Pot : 1
86.98737 : Inventory : Pick : Pot : 1
87.10983 : Inventory : Pick : Pot : 1
87.38734 : Inventory : Pick : Pot : 1
88.86796 : Checkpoint : 3 : MainQuest2POE
91.90846 : LevelEnd
94.68978 : LevelStart : MainQuest2
94.76766 : Checkpoint : 0 : InitialSpawn
97.14484 : PlayerDeath
97.14591 : Inventory : UnEquip : Sword : 1
97.20113 : Checkpoint : 10 : Flag1
98.48672 : RespawnStarted
```

F

Random Trees

```
path_choice_3_s < 0.05 : 0 (12/0)
path_choice_3_s >= 0.05
|   path_choice_3_m < 0.5
|   |   path_choice_5_s < 0.5 : 0 (1/0)
|   |   path_choice_5_s >= 0.5 : 1 (5/0)
|   path_choice_3_m >= 0.5
|   |   survivor_challenge < 0.82 : 0 (1/0)
|   |   survivor_challenge >= 0.82
|   |   |   path_choice_5_s < 0.68 : 1 (1/0)
|   |   |   path_choice_5_s >= 0.68 : 0 (4/0)
```

Figure F.1: The Random Tree which provided the best results in 10-fold cross validation for the Survivor dataset.

```
seeker_challenge < 0.25 : 0 (8/0) [5/0]
seeker_challenge >= 0.25 : 1 (8/0) [3/0]
```

Figure F.2: The Random Tree which provided the best results in 10-fold cross validation for the Seeker dataset.

```
mastermind_challenge < 0.65
|   path_choice_3_m < 0.5 : 0 (9/0)
|   path_choice_3_m >= 0.5
|   |   mastermind_challenge < 0.3 : 0 (2/1)
|   |   mastermind_challenge >= 0.3 : 0 (1/0)
mastermind_challenge >= 0.65
|   path_choice_2_m < 0.9 : 1 (3/0)
|   path_choice_2_m >= 0.9 : 1 (9/2)
```

Figure F.3: The Random Tree which provided the best results in 10-fold cross validation for the Mastermind dataset.

```
path_choice_4_d < 0.8
|   daredevil_challenge < 0.72 : 0 (6/0)
|   daredevil_challenge >= 0.72
|   |   conqueror_challenge < 0.5 : 0 (1/0)
|   |   conqueror_challenge >= 0.5
|   |   |   path_choice_2_d < 0.85 : 0 (1/0)
|   |   |   path_choice_2_d >= 0.85
|   |   |   |   path_choice_4_d < 0.03 : 0 (1/0)
|   |   |   |   path_choice_4_d >= 0.03 : 1 (1/0)
path_choice_4_d >= 0.8
|   daredevil_challenge < 0.5
|   |   conqueror_challenge < 0.33
|   |   |   path_choice_2_d < 0.5 : 0 (2/1)
|   |   |   path_choice_2_d >= 0.5 : 1 (2/0)
|   |   conqueror_challenge >= 0.33
|   |   |   path_choice_2_d < 0.03 : 0 (1/0)
|   |   |   path_choice_2_d >= 0.03 : 1 (4/0)
|   daredevil_challenge >= 0.5
|   |   path_choice_5_d < 0.05 : 1 (1/0)
|   |   path_choice_5_d >= 0.05
|   |   |   conqueror_challenge < 0.5
|   |   |   |   path_choice_5_d < 0.55 : 0 (1/0)
|   |   |   |   path_choice_5_d >= 0.55 : 1 (1/0)
|   |   |   conqueror_challenge >= 0.5 : 0 (2/0)
```

Figure F.4: The Random Tree which provided the best results in 10-fold cross validation for the Daredevil dataset.

```
achiever_challenge < 0.65
|   coins_collected < 0.15
|   |   achiever_armor < 0.38 : 0 (5/0)
|   |   achiever_armor >= 0.38 : 1 (2/0)
|   coins_collected >= 0.15 : 0 (9/0)
achiever_challenge >= 0.65
|   pots_collected < 0.36 : 0 (1/0)
|   pots_collected >= 0.36 : 1 (7/0)
```

Figure F.5: The Random Tree which provided the best results in 10-fold cross validation for the Achiever dataset.