Comparative Performance Analysis of AI Algorithms in Competitive Pokémon Showdown Battles

Nathan Bonavia Zammit

Supervisor: Dr. Kristian Guillaumier

June 2023

Submitted in partial fulfilment of the requirements
for the degree of B.Sc. IT (Hons.) Artificial Intelligence .

L-Università ta' Malta
Faculty of Information &
Communication Technology

Abstract

In this Final Year Project, we investigate Artificial Intelligence bots that play the online strategy game Pokémon Showdown, a web-based Pokémon battle simulator. We first explain the mechanics of Pokémon Battling and how it has evolved throughout the years. We then propose two algorithms that are be assigned to a bot respectively. These bots compete on Pokémon Showdown which acts as an optimal environment to allow these bots to compete against online players. A set number of battles will take place and will be recorded. The comparison of the two algorithms is done using the Elo rating that the platform provides. We also look at the win rate of these bots, as well as the advantages and disadvantages they encountered, along with proposals for future work. Through this, we are able to understand the biggest challenges that each of the algorithms encounters and which one of them performed the best and why.

The two algorithms used are a Minimax algorithm, a search algorithm that looks at future possible turns and prioritises always picking the safest route available, and a Q-Learning algorithm that takes the results of previous battles and tries to learn after each one. These bots are then compared to each other along with two other bots which are referred to as the 'simple' bots. These bots will be a random bot that picks a move on a random basis without any thought behind it, and a high-damage bot that always tries to pick the highest damaging move, regardless of the actual battle taking place. These bots acted as baselines for other algorithms.

The Minimax bot achieved a high win rate in the earlier games but struggled after finding harder opponents which affected its ability to predict the safer routes, whilst the Q-Learning bot plateaued as it did not have any knowledge at the beginning, and then dropped drastically in results. The simple bots were used to showcase the effectiveness of the Minimax and Q-learning bots in comparison since they had very low win rates and Elo ratings, although the Q-learning bot achieved similar if not worse results. After our implementation, we evaluated the performance of the two primary AI algorithms in the context of competitive Role-Playing Game (RPG) game play, specifically in Pokémon battles. Our analysis determined that a search algorithm like Minimax had a higher success rate than a reinforcement learning algorithm like Q-Learning since they were competing in an uncontrolled online environment. Although neither of the two had any huge success in the competitive scene, Minimax struggled less when it came to adapting to various different online users and therefore led to a higher win rate. On the other hand, Q-Learning struggled in the early stages since it had no prior knowledge whatsoever, and since after losing it kept being paired with players with lower Elo ratings, it never manages to improve.

Acknowledgements

This work could not have been accomplished without the outstanding contributions of Guangcong Luo, Kirk Scheibelhut, and the entire development team behind the Pokémon Showdown online platform and battle simulator. Their dedication and expertise have been instrumental in enabling the research presented in this project.

I would also like to express my sincere gratitude to Dr. Kristian Guillaumier for his constant guidance and support throughout this final year project.

Additionally, we would like to express our gratitude to Nintendo and Game Freak for their creation and continued development of the Pokémon franchise. Their efforts have brought joy to millions of people around the world and inspired innovative research in the field of artificial intelligence and gaming.

Contents

Abstract i

Acknowledgements ii

Contents iv

List of Figures v

1 Introduction 1
- 1.1 Aims & Objectives 1
- 1.2 Overview of Pokemon Showdown 2
- 1.3 Why The Problem is non-Trivial 2
- 1.4 Overview of Methodology 3
- 1.5 Overview of Solution 4
- 1.6 Summary of Results 5

2 Background 7
- 2.1 Mechanics of Pokemon Battles 7
- 2.2 The Platform Used - Pokemon Showdown 11
- 2.3 Applications in Other Areas 12

3 Literature Review 14
- 3.1 Existing AI Bots in Games 14
- 3.2 Game Theory in AI: The Minimax Algorithm 15
- 3.3 Reinforcement Learning in AI: Q-Learning 16
- 3.4 Rationale for Using Minimax and Q-Learning 16
- 3.5 Applicability of Search Strategies in Large State Spaces 17
- 3.6 Summary 17

4 Methodology 19
- 4.1 How Battles are Won 20
- 4.2 States in Machine Learning 20
- 4.3 Pathological Game Trees 21

4.4 Summary 22

5 Implementation **23**
5.1 Minimax Algorithm 23
5.2 Q-Learning Algorithm 23
5.3 Simple Bots 24
5.3.1 Max Damage Bot 24
5.3.2 Random Bot 25
5.4 Libraries 25
5.4.1 WebSocket API 25
5.5 Summary 26

6 Evaluation **28**
6.1 Results 29
6.2 Summary 32

7 Conclusions **34**
7.1 Future Work 35
7.2 Closing Remarks 36

List of Figures

Figure 6.1 Algorithm Results 30

1 Introduction

Since the beginning of artificial intelligence, a substantial portion of research has been dedicated to the development of intelligent agents for various games. While considerable progress has been made in deterministic games with perfect information, such as Chess [1] and Go [2], there has been limited exploration of games like Pokémon that involve a significant random component and imperfect information.

As the top-earning media franchise in history, Pokémon has motivated enthusiasts for years to develop smart agents that can effectively rival human competitors [3] [4]. Dating back to 1995, the franchise started off with the first-ever Pokémon games, Pokémon Red and Blue, which debuted on the Nintendo Game Boy Platform. The concept of these games was simple, explore a vast game world full of creatures known as Pokémon. Players would collect, train, evolve, and trade these creatures along with battling against other players using a team of Pokémon. Since then, over 70 different Pokémon games have been released, which together with merchandise, movies, TV shows, cards, and more, have grossed over $90 billion in total revenue. The aforementioned games range from what is known as the core series [5] and spin-offs [6]. For this Final Year Project, we will be focusing on the former as these are the main games that contain the main feature we will be focusing on, competitive single battles.

1.1 Aims & Objectives

Aims:

- Study the behaviour of search Algorithms and reinforcement learning algorithms in the complex and unpredictable environment of Pokémon Showdown to determine which algorithm performs the best in said environment.
- Examine the performance of simpler gameplay algorithms such as a Random Bot in the same context in order to establish a baseline for our evaluation.
- Compare the performance of search algorithms and reinforcement learning algorithms in a competitive, online environment.

Objectives:

- Implement Minimax and Q-Learning bots to act as our main two algorithms which will be compared and looked into further, along with a bot that maximises damage and a bot that always picks a random move to act as baselines.

- Compare the win-to-loss ratios for each algorithm, and analyse the successes and failures in their performances.

1.2 Overview of Pokemon Showdown

On the 18th of November 2022, the first ninth generation core series games were released under the names of Pokémon Scarlet and Pokémon Violet. These brought with them 107 different new Pokémon and a new game mechanic known as Terastallize which changed the dynamic of the games completely. For this reason, we focused on the 8th generation of Pokémon Games, Pokémon Sword, and Pokémon Shield as at the time the online competitive scene for the ninth generation had not fully settled and too many changes were taking place, thus we focused on the eighth generation which had settled years ago. Whilst, each game brings new changes, the concept of two players connecting locally and battling each other has remained unchanged. In 2006 players could also do so through the Internet.

In 2009, the first-ever Video Game Championships (VGC) was organised by the game publisher, Nintendo. In this event, the top players from all around the world are invited to compete in a tournament of Pokémon battles to find the best of the best and earn the title of World Champion. This was what led to the creation of professional Pokémon players. Whilst the core gameplay mechanics were designed to be easy to pick up and play, the concept of high-level battling was not as easy for professional players to master since the battle mechanics are relatively complicated. The Pokémon community itself has grown massively, to the point where various websites, which are run by fans, provide further analysis of the game mechanics in order to aid players in any shape or form, whether they be amateurs or professionals [7] [8] [9].

1.3 Why The Problem is non-Trivial

The non-trivial nature of our Final Year Project lies in the fact that it addresses the challenge of creating and comparing AI bots for Pokémon battles, a complex and uncertain environment. By studying and comparing the performance of different AI algorithms, such as Minimax and Q-Learning, we aim to better understand the strengths and weaknesses of these approaches in handling strategic decision-making under conditions of uncertainty and imperfect information.

The problem we are tackling has broader implications beyond the realm of Pokémon battles. A robust AI agent that excels in managing teams under highly uncertain conditions can potentially find applications across various domains, including team management in dynamic and high-risk environments. For instance, such an AI agent could be adapted to manage teams of doctors, robots, or employees in situations where circumstances change rapidly and unpredictability is the norm, such as in pandemic-stricken regions or war zones. Developing an AI agent capable of thriving in these conditions can significantly impact the effectiveness of operations and decision-making in these challenging contexts.

Moreover, our findings can also be applied to other popular Japanese Role-Playing Games (J-RPGs) that share similar battle mechanics with Pokémon, such as Final Fantasy. Many J-RPGs emphasise the importance of teamwork and strategic thinking under conditions of extreme uncertainty to achieve victory. By understanding the performance of different AI algorithms in the context of Pokémon battles, we can gain insights into how these algorithms may be adapted and applied to other J-RPGs, potentially enhancing the gaming experience and fostering the development of more sophisticated AI agents for these games.

1.4 Overview of Methodology

Our methodology centres around devising an effective solution to managing the complex and uncertain decision-making landscape of Pokémon battles through the application of artificial intelligence. We propose, implement, and evaluate a variety of AI bots, each driven by different decision-making strategies, with the ultimate goal of identifying the most effective approach for managing uncertainties inherent in such strategic environments.

Our first strategy involves developing a bot based on the Minimax algorithm [10], a well-established adversarial search algorithm widely used in two-player turn-based games. The Minimax algorithm assists the bot in making informed decisions by forecasting future possibilities and opting for the move that minimises the maximum potential loss. To make this process computationally efficient, we employ Alpha-Beta pruning to curtail the exploration of non-promising branches in the search tree [11].

Our second approach involves utilising Q-Learning, a model-free reinforcement learning algorithm [12], to create a bot that learns from its interaction with the environment. The Q-Learning bot adapts its strategy over time, building a policy that maximises expected rewards based on the feedback received.

In addition to these sophisticated AI approaches, we develop simpler rule-based bots to serve as a control group. These include the Most Damage bot, which makes decisions primarily based on the potential damage inflicted, and the Random bot, which selects actions randomly without considering any strategic implications.

To empirically evaluate these bots and compare their performances, we integrate them into the Pokémon Showdown online platform using its WebSocket API. We facilitate these bots to engage in Generation 8 Random Battles against human players and track their performance over time using the Elo rating system [13].

The crux of our proposed solution is to understand the implications of these varied AI approaches in the Pokémon battle environment. By comparing the performance of bots driven by different strategies, we aim to discern which strategy best navigates the uncertainties and complexities of this strategic environment. Our analysis from this study could provide the foundation for building an AI agent capable of making effective decisions in Pokémon battles, and potentially other similar environments, marking a significant advancement in the application of AI in gaming.

1.5 Overview of Solution

Our project's solution aims to advance the understanding and application of artificial intelligence in the realm of Pokémon battles by designing and implementing various AI bots, each driven by a unique decision-making strategy. These bots interact with real human players in an online environment, Pokemon Showdown, where the uncertainties and complexities of the game challenge the decision-making capabilities of the AI.

The Minimax bot, the first of our developed solutions, employs the Minimax algorithm complemented by Alpha-Beta pruning. This adversarial search algorithm allows the bot to predict the consequences of its decisions, helping it choose the move that minimises the potential loss in the worst-case scenario. This approach is particularly suited for the deterministic parts of Pokémon battles, providing a strategic edge in making well-informed decisions in a turn-based adversarial context.

Secondly, we implement a Q-Learning bot utilising the principles of reinforcement learning. This model-free algorithm enables the bot to learn from its interactions with the environment and adapt its strategy based on the feedback received. Over time, the Q-Learning bot constructs a policy that maximises expected rewards, effectively learning from its past experiences to enhance its future performance. This approach offers a significant advantage in handling the stochastic aspects of Pokémon battles, enabling the bot to learn and adapt from the uncertain and random elements of the game.

In contrast to the sophisticated AI approaches, we also introduce simpler, rule-based bots: the Most Damage bot and the Random bot. The Most Damage bot prioritises the immediate outcome, choosing the move with the highest potential damage, while the Random bot selects actions at random without any strategic consideration. These bots serve as control groups, providing a baseline to understand and appreciate the benefits and advancements brought about by the more complex AI algorithms.

The evaluation of these bots' performance forms an integral part of our solution. We leverage the Pokémon Showdown platform's automated matchmaking system to pit our bots against human opponents. By recording their performance over multiple battles, we gain insight into each bot's strengths and weaknesses and the effectiveness of their underlying decision-making strategies. We primarily use the Elo rating system, a reliable method to assess a player's skill level, to measure and compare the bots' performances.

In essence, our solution is a comprehensive investigation into the application of AI in the context of Pokémon battles, focusing on both the deterministic and stochastic elements of the game. Our bots, powered by diverse decision-making strategies, offer invaluable insights into the potential of AI in this challenging environment. Our comparative analysis of their performance provides a foundation for future research and development in this field, contributing to the broader goal of advancing AI in gaming.

1.6 Summary of Results

The culmination of our efforts in this project has resulted in a plethora of insightful findings, furthering our understanding of AI's potential and limitations within complex and unpredictable environments, particularly Pokémon battles. We present here an initial glimpse into our results, highlighting the broader implications and patterns observed throughout our research.

Our exploration of varying decision-making strategies revealed compelling differences in their performances within the Pokémon showdown environment. The Minimax bot, rooted in adversarial search algorithm strategies, demonstrated a strong competency in deterministic contexts, underscoring the value of anticipatory, strategic decision-making in turn-based games.

Contrastingly, the Q-Learning bot, which operates based on principles of reinforcement learning, displayed a marked learning curve. It showcased an ability to glean from past experiences and iteratively refine its future strategies, hinting at the potential of model-free reinforcement learning in these complex, uncertain environments.

Simultaneously, the performances of the simpler Most Damage and Random bots offered valuable insights. These rule-based bots provided a necessary frame of reference to understand the intricacies and effectiveness of more advanced algorithms. Their basic decision-making approaches demonstrated the pros and cons of relying solely on immediate outcomes or purely random selections.

Despite the sophistication of the AI algorithms, the bots faced both successes and setbacks in their battles against human opponents. This mixture of outcomes illustrates the inherent complexities and unpredictability of the Pokémon battle environment, reiterating the challenges posed to AI in situations of imperfect information and uncertainty.

Employing the Elo rating system allowed us to quantify our bots' performances, offering an objective measure of their competence over time. The comparison of different decision-making strategies and their effectiveness provided through this rating system emerged as an illuminating facet of our research.

Our initial results shed light on the profound implications of this project, raising critical questions about strategic decision-making, the versatility of AI algorithms, and the dynamics of Pokémon battles. The remainder of this report will delve further into the depths of these fascinating results, offering a comprehensive exploration of the intersection of AI and the gaming landscape.

2 Background

2.1 Mechanics of Pokemon Battles

This section sheds light on the principles governing Pokémon battles. We initiate our discussion with the gameplay of battles, proceed to delve into the attributes of the creatures called Pokémon, and wrap up with an analysis of the structure of battle states. The exploration of the game world and the collection, training, evolution, and trading of Pokémon will not be covered, as these elements are beyond the purview of this study. Our concentration is on competitive battles that emphasise victory, rather than in-game battles aimed at training or capturing wild Pokémon. The information presented pertains to the core series games spanning generations 6 to 9. For a deeper understanding, readers can consult online sources [7] [8] [9].

Each battle involves two opposing sides, with one, two, or three Pokémon actively participating from each side at any given time. Additionally, each side has reserve Pokémon that can be called upon to replace active ones when necessary. Battles are classified as Single, Double, or Triple, based on the number of active Pokémon on each side. Typically, both sides have an equal number of active Pokémon. This study concentrates on Single battles, but the approach can be extended to encompass the other categories as well. The total number of Pokémon per team, including active and reserve members, is determined by the specific battle rules and cannot exceed six. Our research emphasises battles with the maximum allowable Pokémon per team.

The game employs a turn-based battle system. At the beginning of each turn, both sides must simultaneously choose their actions without prior knowledge of their opponent's choices. Actions are only executed once both sides have made their selections, and the order of execution is detailed below.

1. **Make an active Pokémon use a move.** Each Pokémon on the field may use one move per turn.

2. **Switch out your active Pokémon with a reserve Pokémon.** A Pokémon is allowed to switch with another at least once per turn.

3. **Use an item from the item bag [14].** There are various items a player is able to use during battles. Items are also used when it comes to other aspects of the games, such as exploration of the games, and training of Pokémon.

4. **Flee the battle.** This is often used when the opponent is a wild Pokémon, however, in an online competitive setting it is also used as a forfeit button.

In competitive battles only the first two, and in rare occurrences the last option, are possible. In these battles, players are not permitted to access their bags; however, each Pokémon can hold an item. The sequence in which selections are carried out is determined by the game mechanics. With a few uncommon exceptions, Pokémon switching occurs before any moves are executed. The sequence of move execution relies on several Pokémon properties, which will be discussed later. The three most crucial factors are the speed statistic, ability, and move effect. Players are aware of their own Pokémon's properties but remain uninformed about their opponent's Pokémon properties. Consequently, players must make educated guesses regarding this concealed information to figure out which Pokémon will act first.

When a player swaps an active Pokémon for another one, provided the active Pokémon is still capable of battling, the newly switched-in Pokémon may take a hit upon entering the battle arena without having a chance to retaliate if the opponent opts for an attack. Consequently, swapping Pokémon without being compelled to do so carries inherent risks and should ideally be done only if the anticipated opponent move will inflict minimal damage or if the opponent is also predicted to switch their Pokémon.

Victory in a battle is achieved by the player who first depletes the health of all opposing Pokémon. Team balance must be taken into account since a team consists of multiple Pokémon, and the player must decide whether it is better to have a single fully healthy Pokémon or several injured ones. The objective is not merely to inflict as much damage as possible on the enemy but to ensure that all their Pokémon eventually faint. This is made more complex by the numerous conditions that can impact the battle, as we discuss later.

Assessing the value of battle states is incredibly complex due to the need to consider team balance, battle conditions, and the uncertainty arising from hidden information, along with a multitude of exceptional cases. Occasionally, reducing an opposing Pokémon's health points without fully depleting them can backfire, as this might trigger an ability that doubles its attack power. In many instances, even professional players engage in heated debates over which battle state is more favourable.

Having looked into the details of how Pokémon battles are played, let's now look into detail the properties of Pokémon.

1. **The species:** Every Pokémon belongs to one of the over 1000 possible species [15] [16]. Ranging from Bulbasaur at #0001 to Iron Leaves at #1010, the species of the Pokémon is important as it impacts the battle in various ways.

2. **The level:** Every Pokémon has a level range from 1 to 100 [17]. The level may not have a direct effect on the battle itself, however, it is significant as it has a major

impact. When a Pokémon is higher levelled:

   a. It has higher statistics
   b. It is able to learn stronger moves
   c. It can even evolve into a stronger species.

3. **The type(s):** Every Pokémon has up to one or two types [18] out of: Normal, Fire, Water, Grass, Electric, Ice, Fighting, Poison, Ground, Flying, Psychic, Bug, Rock, Ghost, Dragon, Dark, Steel, and Fairy. For example, Pikachu is an Electric type, whilst a Pokémon like Venusaur is Grass and Poison type. Types play a big part in Pokémon battles as they increase and decrease the effectiveness of moves. Let's take, for instance, the Fire and Water types. If a Fire type Pokémon is hit with a Water type move then double the damage is dealt, whilst if a Water type Pokémon is hit with a Fire move, then half the damage is dealt. In some cases, moves can be four times effective. For example, if a ground and flying type Pokémon is hit with an Ice type move, then the damage is multiplied by four. A type chart [19] can be looked into to better understand this.

4. **The statistics:** Every Pokémon has a set of statistics, these being: Hit Points (HP), Attack, Defence, Special Aattack, Special Defence and Speed [20]. These are all determined by a formula [21] which takes into account the Species, Level, Nature [22], Individual Values (IV) [23] and Effort Values (EV) [24]. We will not be looking into the last three parameters as they only impact the statistics and not the battles themselves. Once the formula calculates the value of a statistic, this is can be modified even further through the use of items, abilities, move effects and battle conditions.

5. **The ability:** Every Pokémon can have one ability [25] from three possible abilities, depending on the species. For example, Pikachu has the option of two separate abilities, Static and Lightning Rod, whilst some Pokémon such as Gigalith has three abilities, Study, Sand Stream and Sand Force. The ability impacts directly the battle in many ways since it can modify the statistics, status the opponent, change the effectiveness of moves, change the battle conditions and more. At the time of writing there over 290 abilities in total [26].

6. **The item:** Every Pokémon, can optionally hold of over 1,500 items which are available in the game [27] [28]. Examples of these items are Leftovers, which heal a small percentage of health after each turn, and Choice Band, increase the Pokémon's Attack statistic but only allows it to use one move. Items can have numerous different effects in battle, ranging from healing the Pokémon as mentioned earlier, to even forcing an opponent to switch out their Pokémon.

7. **The move(s):** Every Pokémon knows one to four moves [29]. The pool of possible moves depends on the species as some Pokémon can only know 1 move whilst some are able to learn over 100 different moves. For example, Pikachu is able to learn Thunderbolt, but can learn Flamethrower. Moves have various properties: type, category, damage, accuracy, power points (PP) and sometimes even an additional effect. A non-deterministic formula [30] takes into account the properties of the move along with the properties of the attacking and defending Pokémon and determines the damage. The move category is either Physical, Special or Status. This determines if the Attack and Defence, or the Special Attack and Special Defence statistics will be used when it comes to the damage calculation. Power points refers to the maximum number of times a move can be used in a battle. Some move effects cause additional status effects such as healing the user, giving the opponent a status effect, or even changing the battle conditions. At the time of writing, there are 900 moves in the games [31].

Now that we have looked into the key properties of the creatures known as Pokémon, let's look into the format of the battle states. Since there is a large amount of hidden information in the game, only the computer running the battle software can fully observe the battle state. These states include:

1. **The player teams:** The team of every player is stored in the battle state. Depending on the battle rules, the properties of the opponent's Pokémon, including in many cases the species, are unknown unless they are revealed. Every player tries to figure out as much hidden information about the opponent's team throughout the battle.
2. **The battlefield conditions [32]:** These can be categorised as follows:
    a. **Two-sided conditions:** These influence the entire battlefield and typically persist for several turns. For example, weather changes like rain can enhance the strength of Water-type moves and ensure specific moves always hit their targets. Another example is the creation of a psychic energy field called Psychic Terrain, which amplifies Psychic type moves and prevents opponents from utilising moves with higher priority.
    b. The battle-side conditions: These affect only one side of the battlefield. Some persist for a few turns, while others last the entire battle unless negated by an effect. For instance, laying toxic spikes on the enemy's side causes all enemy Pokémon to be poisoned upon being switched in. Another example is generating a tailwind on the ally's side, which doubles the speed statistic of all ally Pokémon for a limited number of turns.

c. The Pokémon conditions: These influence individual Pokémon exclusively. Some endure for a few turns, others persist until the Pokémon is switched out, and some last the entire battle unless negated by an effect. One example is altering a Pokémon's status to frozen, burned, or paralysed. Another involves increasing or decreasing any of a Pokémon's statistics. It is even feasible to temporarily change a Pokémon's form during a battle.

2.2 The Platform Used - Pokemon Showdown

The core series Pokémon games have been exclusively available on Nintendo platforms, including the Nintendo Switch. Until recently, the games made it challenging to engage in ranked battles with other players online. In these battles, players are ranked on a global leader board, using a ranking system similar to that of professional Chess players. Moreover, many online features are only accessible for a year or two after a game's release, prompting players to purchase the latest version nearly every year to continue participating in online battles. The enthusiasm of Pokémon fans, combined with these challenges, led to the development of a fan-made Pokémon battle simulator called Pokémon Showdown.

Initiated in 2011 by Guangcong Luo, Pokémon Showdown is currently an open-source project with over 20,000 commits and more than 300 contributors on GitHub [33]. The project effectively simulates the highly complex Pokémon battle system, with only rare bugs affecting its near-perfect performance. It does not simulate aspects such as exploration of the game world or Pokémon collection, training, evolution, and trading. An online platform was built around the simulator, enabling players to engage in ranked battles. Accessible via a webpage (https://play.pokemonshowdown.com/), players' rankings are based on their wins and losses, determined by the Elo rating system. Players are matched with opponents according to their Elo ratings. If either player requests it, the platform imposes a strict time limit of a few seconds per decision, which can pose challenges for AI agents.

Pokémon Showdown offers battles in several different formats [34]. Said formats have different rules and challenges for players. The biggest differences between each format are the following:

1. **Team selection:** At the start of a battle, each player must select a team of Pokémon to use or be provided with a team of randomly generated Pokémon. Building a strong Pokémon team is no easy task, whilst it is significantly harder to master every possible random team that is available in the Random Battle format.

2. **Opponent Team Preview:** At the start of every battle, the species property of every opponent Pokémon is either revealed or kept hidden until said Pokémon is

used at least once in battle. Whenever a Pokémon is kept hidden, battles are significantly harder as players have to make assumptions about figuring out hidden information.

3. **Number of Active Pokémon:** Pokémon battles are categorized as Single, Double, or Triple battles, depending on the number of active Pokémon per side.

4. **Number of Pokémon per team:** Pokémon teams can have a maximum of six Pokémon, one active, and then the rest are reserves.

5. **Restrictions:** A battle format can have several restrictions, including banned Pokémon, moves, items, abilities, and more.

6. **Generation:** Every generation of Pokémon games differ in the available Pokémon, moves, items, and abilities, as well as a battle mechanic. All generations from 1 to 9 are available on the platform.

The online platform features two widely popular battle formats: ”Random Battles” and ”OverUsed (OU) Battles” [35]. In Random Battles, players are assigned a random team at the beginning of the battle, and there is no preview of the opponent’s team. In OU Battles, players choose their teams, and a preview of the opponent’s team is provided. Both formats are Single battles, where players have one active Pokémon at a time, and each player has a team of six Pokémon. While there are differences in the restrictions of the two battle formats, they will not be discussed here due to space limitations. In this work, we focus on the 8th generation of core series Pokémon games and the Random Battles format, which is considered more popular and challenging than the OU Battles format.

The simulator and the online platform are essential infrastructure components for developing a Pokémon AI agent. Without the simulator, an AI cannot employ look-ahead strategies. To date, Nintendo has not provided an API for programmatic interaction with their Pokémon games. As a result, the only way for an AI agent to play against human players on a Nintendo platform is by analysing the video captured from the console screen. The online platform based on Pokémon Showdown addresses this issue, offering a WebSocket API that AI agents can use to receive a stream of battle events and respond with battle commands.

2.3 Applications in Other Areas

Our work in comparing AI algorithms for Pokémon battles has significant applications beyond gaming, extending to real-world scenarios such as disaster response and recovery, and supply chain management. For instance, efficient coordination of rescue teams, drones, or robots during natural disasters requires strategic decision-making under uncertain conditions, which can benefit from AI algorithms adept at handling such environments. Similarly, managing global supply chains involves navigating uncertainties like fluctuating demand, political instability, or transportation disruptions, and AI algorithms that excel in these situations could help optimise operations and minimise disruptions. In addition to these real-world applications, our findings can also be applied to other popular Japanese Role-Playing Games (J-RPGs), such as Final Fantasy, which share similar game mechanics with Pokémon and emphasise the importance of teamwork and strategic thinking under extreme uncertainty. Our project's focus on comparing AI bots in Pokémon battles holds broader relevance and impact across various domains, demonstrating the wide-ranging applicability of our work.

3 Literature Review

3.1 Existing AI Bots in Games

The application of artificial intelligence in gaming has been a prevailing focus within both the AI research community and the game industry. Complex multiplayer games, such as Pokémon Showdown, provide a rigorous testing ground for the development and application of AI strategies. Over the years, a variety of AI bots have been developed and deployed in such environments, each employing distinct techniques and demonstrating varying degrees of success.

In the case of Pokémon Showdown, a complex multiplayer game with a vast array of possibilities for each move, several AI bots have been developed. For instance, Teye et al. [36] used reinforcement learning to train a bot for Pokémon battles. This bot continually updated its understanding of the most advantageous actions based on past experiences, demonstrating the potential for machine learning in such complex environments. Similarly, Calvo and Gutierrez [37] constructed a rule-based AI bot, leveraging decision trees to predict the opponent's moves and plan accordingly.

Beyond Pokémon Showdown, there are various notable instances of AI bot deployment in other games. Perhaps one of the most famous is IBM's Deep Blue, which defeated a reigning world chess champion, Garry Kasparov, in 1997 [38]. This was a significant milestone in the history of AI in games, as it showcased the potential of AI to handle deterministic games with perfect information, utilising a brute-force algorithm to evaluate millions of possible positions.

On the more complex end of the spectrum, we have the board game Go. Despite its simple rules, Go has a significantly larger state space than chess, making it a much more challenging arena for AI. Nevertheless, DeepMind's AlphaGo, which employs a combination of Monte Carlo Tree Search and deep neural networks, defeated a human world champion in 2016 [39].

AI bots have also been successfully applied in video games. For example, in the popular multiplayer online battle arena game Dota 2, OpenAI Five utilised a reinforcement learning algorithm to predict the outcomes of various actions, leading it to defeat human teams at the 2018 International Dota 2 Championship [40].

Similarly, in StarCraft II, a real-time strategy game characterised by hidden information and rapidly changing game states, DeepMind's AlphaStar leveraged a variant of reinforcement learning and self-play to reach the top 0.15% of human players on the official online ladder in 2019 [41].

In summary, the success of AI bots across a wide range of games, from deterministic board games like chess to complex multiplayer video games like Dota 2 and StarCraft II, as well as Pokémon Showdown, underscores the potential of AI to grasp intricate game strategies, handle vast game states, and make informed decisions amidst uncertainty. The various implementations of AI in these games form a firm foundation upon which we build our investigation of AI bots in Pokémon Showdown, employing both the Minimax and Q-Learning algorithms [42].

3.2 Game Theory in AI: The Minimax Algorithm

The Minimax algorithm is deeply rooted in game theory and is considered a standard solution to two-player zero-sum games. This algorithm has found widespread use in artificial intelligence, particularly for deterministic games that involve perfect information, like chess or tic-tac-toe [43].

The concept of the Minimax algorithm centres around a player maximising their possible minimum gain (or minimising their maximum loss). It operates by exploring the game tree to evaluate potential actions and their consequences, following a depth-first search strategy. At each decision point (or node) in the game tree, the Minimax algorithm anticipates the opponent's best response to every possible move. It then chooses the action that will result in the maximum possible benefit, assuming optimal play from the opponent [44].

One of the most renowned applications of the Minimax algorithm was in IBM's Deep Blue, the first artificial intelligence to defeat a reigning human world chess champion, Garry Kasparov, in 1997 [38]. This event served as a significant milestone in AI, demonstrating the potential of the Minimax algorithm in games that require intricate strategic decision-making.

Despite its success in chess, a deterministic game with perfect information, applying the Minimax algorithm to games like Pokémon, which involve hidden information and uncertain outcomes, presents a unique challenge. This challenge emerges from the need to deal with a large game tree due to the considerable complexity and vast state space in Pokémon battles [45].

Several enhancements to the basic Minimax algorithm have been proposed to deal with these complexities, including Alpha-Beta pruning [11], which helps in managing the computational load by eliminating branches in the game tree that do not need to be searched because they cannot possibly influence the final decision. Other refinements such as the Expectimax algorithm [46] incorporate elements of chance into the Minimax framework, which can be useful in managing the inherent randomness in Pokémon battles. The combination of such techniques offers the potential to harness the power of the Minimax algorithm effectively for Pokémon Showdown.

3.3 Reinforcement Learning in AI: Q-Learning

Q-Learning, a form of reinforcement learning, is a model-free algorithm that seeks to learn the value of being in a state and taking a specific action in that state [47]. This learning is accomplished by iteratively updating the estimated Q-values using the observed rewards and the maximum predicted Q-values for the next state. The primary goal of Q-Learning is to find the optimal policy that maximises the cumulative reward over all steps, starting from the current state.

A key characteristic of Q-Learning that has led to its successful deployment in various AI applications is its ability to learn from its past actions and their associated rewards. This feature is particularly beneficial in dynamic and stochastic environments, as it allows the algorithm to adapt its strategy based on new experiences.

One of the most significant successes of Q-Learning in AI was demonstrated by DeepMind's AlphaGo, the first AI to defeat a human world champion in Go, a highly complex board game [39]. AlphaGo used a variant of Q-Learning combined with deep neural networks, known as Deep Q-Learning [48]. This combination allowed AlphaGo to handle the vast state space in Go, which is many orders of magnitude larger than that of Chess.

In the context of Pokémon battles, Q-Learning's adaptability and learning mechanism present an attractive solution for managing the inherent randomness and complexity of the game. However, it should be noted that traditional tabular Q-Learning struggles when dealing with large state spaces due to the ”curse of dimensionality” [49]. To overcome this challenge, approaches such as function approximation [50] and deep reinforcement learning [51] have been proposed. These techniques allow the learning of Q-values even in situations with high-dimensional or continuous state spaces, thus potentially making them applicable to Pokémon Showdown.

3.4 Rationale for Using Minimax and Q-Learning

The decision to employ both the Minimax algorithm and Q-Learning in our approach to Pokémon battles was informed by several considerations.

Firstly, the Minimax algorithm, recognised for its efficacy in strategic decision-making and predicting opponent responses, offers considerable potential in enhancing the bot's performance in Pokémon battles. The fundamental strategy of Minimax, which assumes that the opponent will always make the move that is most detrimental to the AI's position, adds a depth of strategic foresight that is crucial in a tactical game like Pokémon.

However, Pokémon battles involve an element of uncertainty and randomness, characteristics that can potentially challenge the effectiveness of the Minimax algorithm, originally designed for deterministic games with perfect information. Here, Q-Learning presents a viable complement. Its foundation on reinforcement learning, learning from past actions, and adaptability to evolving game dynamics offers a way to handle the inherent randomness in Pokémon battles effectively.

The combination of these two techniques, the strategic anticipation of Minimax and the adaptive learning of Q-Learning, is aimed at addressing the complex and stochastic nature of Pokémon battles. The amalgamation of these techniques is designed to create an AI bot that can handle both the strategic depth and inherent uncertainty present in Pokémon battles, learning from its past experiences and making strategic decisions based on the anticipation of the opponent's actions [52].

3.5 Applicability of Search Strategies in Large State Spaces

The exploration of search strategies is a cornerstone in the design of AI for games. These strategies, often rooted in traversing a game's state space, have found success in many traditional board games like Chess and Go. However, when faced with a game like Pokémon, with its vast state space and inherently stochastic dynamics, the application of these conventional search strategies warrants careful consideration.

A typical search strategy like depth-first or breadth-first search, while straightforward and effective in many cases, may struggle to handle the enormous state space in Pokémon battles effectively. This immense state space is a result of the large number of Pokémon, each with its unique attributes, moves, potential held items, and the various status effects and battle conditions that can occur during a battle.

While the exploration of the state space remains a fundamental aspect of our approach, the sheer size and complexity of the state space in Pokémon battles necessitate a nuanced approach. Thus, we aim to balance exhaustive exploration with strategic foresight and learning, thereby leveraging the strengths of both Minimax and Q-Learning. Our approach aims to effectively navigate the vast and complex state space while ensuring the bot's adaptability and strategic competence [53].

3.6 Summary

This Literature Review chapter analysed the current state of artificial intelligence (AI) in games, focusing on AI bots in Pokémon Showdown and relevant strategies. It examined the application of reinforcement learning and rule-based methods in gaming AI, showcasing examples from various gaming platforms including Hearthstone and StarCraft II. Two pivotal AI techniques, the Minimax algorithm and Q-Learning, were explored in depth, along with their potential benefits and challenges when applied to games with hidden information, large state spaces, and inherent randomness like Pokémon. The chapter underscored the strategic value of combining Minimax's predictive capability with Q-Learning's adaptability to enhance the bot's performance in Pokémon battles. Furthermore, the discussion highlighted the necessity for a nuanced approach to navigating the vast state space of Pokémon, balancing exhaustive exploration with strategic foresight and learning. In summary, the chapter provided an encompassing study of AI application in games, emphasising both the complexities and the potential strategies to handle such environments.

4 Methodology

In the subsequent section, we undertake a meticulous exploration of our key findings about the multifaceted game of Pokémon, which forms the underlying foundation of our study. The observations and conclusions derived from this examination serve as pivotal stepping-stones, guiding us in formulating hypotheses about the performance of the two algorithms we have chosen to apply - Minimax and Q-Learning.

These algorithms, each with their distinctive strengths, offer a rich potential to be tapped in the realm of AI for gaming. We have sought to bring these advantages to the fore in the unique context of Pokémon battles, a domain that presents a blend of strategic depth, uncertainty, and dynamic gameplay. Consequently, the insights we gathered about the game and its intricacies played an instrumental role in anticipating the potential of these algorithms and predicting their performance.

This groundwork is not simply a prelude to our experiment, but rather, it constitutes an integral part of our research journey. Understanding the game's mechanics, its players' strategies, the various scenarios that can emerge in the course of a battle, the elements of randomness and hidden information inherent in the gameplay - all these aspects are critical in shaping the AI's approach to Pokémon battles.

It is important to note that the complex nature of Pokémon battles introduces a variety of factors that can influence the algorithms' effectiveness. From the varied abilities of individual Pokémon to the unpredictable manoeuvres of opponents and the changing dynamics of a battle, each component contributes to the overall performance of the algorithms. Consequently, it is these very game-specific conclusions that we draw upon to understand and predict how our proposed AI agents would fare in this unique environment.

By diving deep into the game's mechanics and nuances, we are better equipped to calibrate our AI agents, tailor the algorithms, and anticipate potential challenges. We believe that this intricate understanding of the game, combined with the capabilities of Minimax and Q-Learning, will ultimately contribute to the creation of a formidable AI bot for Pokémon battles.

With this purpose in mind, we delve into the specifications and design considerations that shape our approach, discussing how victories in battles are attained, the significant factors to consider when applying Machine Learning, and the peculiarities of the game trees in Pokémon, among other things. Each of these aspects will be elucidated in the following sections, providing a comprehensive overview of our thought process, decisions, and expectations regarding the AI agents' performance.

4.1 How Battles are Won

After looking into the Pokémon battle mechanics earlier on, it should be clear that in order for a player to succeed in the competitive scene he must memorise a lot of information. They would need to know the properties of over 1000 Pokémon species, the move sets they can learn and what moves do, what items they often carry, and understand the vast number of abilities that are available. Not only that but they must also understand the battle conditions of Generation 8, the generation which this work is based on. Great players understand that victory depends on:

1. **Keeping the team as balanced as possible.** In some scenarios, it is worth taking a short-term loss in order to keep the team as balanced as possible and capable of winning the game in the long run. For example, sometimes it is worth switching in an already damaged Pokémon which has already done its job on the team, whether that be setting up a specific battle condition to assist the team, or taking out a specific threat to the team which otherwise would have led to a complete loss. Or another example would be that it is worth losing a fully healthy Pokémon in order to allow the Water type on the team to survive and then take out the opponent's two Fire types as otherwise there would not be any way the battle could be won. Maximising the probability that the team has a higher chance of winning is every great player's priority. In other words, they want to be sure that they have at least a way to beat all opponent Pokémon with at least one of their Pokémon still standing.

2. **Dealing with the different sources of uncertainty.** These sources are:

   a. Players decide simultaneously and often intentionally to make unexpected moves. As a result of this, opponent prediction is critical.

   b. A large amount of hidden information is present since many of the opponent's properties are unknown unless revealed or inferred.

   c. The damage and the effect of the vast majority of Pokémon moves are non-deterministic. Each move can lead to over 1,000 different states, depending on the output of the Random Number Generation.

4.2 States in Machine Learning

The following are some important characteristics one should look into before applying machine learning to competitive Pokémon Battles.

1. **Complexity of the game mechanics:** The game mechanics of Pokémon battling are substantially more complicated compared to games like Go or Chess. Chess,

for instance, has an estimated game-tree complexity of about 10^{123} [54], while Go, even with its significantly larger board, has an estimated complexity of about 10^{360} [55]. However, the number of possible battle states in Pokémon battles surpasses even these figures due to the numerous species, moves, items, abilities, and conditions that can exist in any given state [56]. These complexities pose a significant challenge for Neural Networks to infer meaningful patterns from the game states. Directly tasking the Neural Networks to evaluate these states may result in sub-optimal performance, as they may struggle to learn the intricate game mechanics indirectly [50].

2. **Uniqueness of battle environments:** Each Pokémon battle presents a unique environment, mainly because player teams are rarely identical, especially in the battle format focused on in this study. This variability is akin to non-stationary environments where the distribution of the states can change over time, presenting significant challenges for machine learning applications [57]. Training Neural Networks exclusively on past battles using simple supervised learning could lead to a lack of representation for the ongoing battle's environment in the training set. This situation could limit the generalisability of the AI agent and its ability to adapt to new, unseen battle configurations [58].

3. **Role of uncertainty and luck:** The Pokémon battle system has inherent uncertainty due to simultaneous move selection, hidden information, and non-deterministic move outcomes. These uncertainties can lead to a broad range of possible states from the same move, contingent on the game's Random Number Generator (RNG) [59]. This makes it challenging to precisely assess the role of skill versus luck in victory outcomes. Traditional reinforcement learning approaches, which reward the AI agent based on the outcome, may not be optimal in such conditions. It's plausible for players to win despite poor decision-making, or lose despite playing optimally, due to the influence of RNG [45]. This suggests the need for more nuanced learning algorithms or reward systems that can better account for the game's inherent uncertainty.

4.3 Pathological Game Trees

Dana Nau [60] [61] identified that in certain games, excessively focusing on predicting numerous future turns could result in inferior decision-making. This observation was termed ”game tree pathology.” After conducting an in-depth examination of the game, we concluded that the game trees possess pathological traits. In Pokémon this is almost omnipresent as when attempting to anticipate an additional turn, we must make assumptions about the opponent's decisions during that turn, estimate the hidden information concerning the opponent's Pokémon attributes, and presuppose the values the RNG will generate. The game tree exhibits pathological characteristics because, as we look further ahead, our assumptions build upon one another, causing the AI to become increasingly detached from reality as it envisions a future that significantly deviates from what will actually transpire. Unfortunately, completely disregarding look-ahead is not feasible in this game. Therefore, we must ascertain the optimal number of turns to look-ahead to achieve a balance between forecasting the future and minimising the problems we have described.

4.4 Summary

This chapter focused on meticulously understanding the underlying game mechanics of Pokémon, with insights derived from this exploration serving as a pivotal basis for implementing and evaluating the chosen algorithms: Minimax and Q-Learning. Recognizing the game's complexities and its inherent blend of strategic depth, uncertainty, and dynamic gameplay was critical in predicting the performance of the algorithms in this unique context.

The first segment detailed the importance of balance in a team and handling uncertainty effectively to achieve victory. This understanding shaped the approach toward the application of machine learning in the complex environment of Pokémon battles. The game's mechanics were contrasted with other complex games like Chess and Go, emphasising the unique challenges posed by the vast number of possible battle states, the distinctness of each battle environment, and the role of uncertainty and luck.

The section titled "States in Machine Learning" delved into the considerations to be accounted for when applying machine learning to Pokémon battles. It discussed the challenges associated with the game's complexity, the uniqueness of battle environments, and the role of uncertainty and luck.

The final section identified the characteristic of game tree pathology within Pokémon battles. The game tree's pathological traits, with increasing assumptions about the future causing the AI to become increasingly detached from reality, were highlighted. This necessitated a balance between forecasting the future and minimising these issues, an aspect deemed crucial in the application of AI to Pokémon battles.

Collectively, this chapter emphasised a deep understanding of the game's mechanics and nuances as integral to calibrating the AI agents effectively, shaping the approach towards implementing the algorithms, and predicting their performance.

5 Implementation

In this section, we delve into the methodologies and thought processes behind the implementation of our Minimax and Q-Learning algorithms. Our goal is to provide sufficient detail so that the reader can grasp the intricacies and challenges faced during the development stage.

5.1 Minimax Algorithm

The Minimax algorithm, a staple in decision-making strategies for two-player games, presents an attractive tool for application in Pokémon battles. Conceptually, it was built with the goal of selecting an optimal move that maximises our AI bot's survivability by minimising the possible maximum advantage gained by the opponent. This dual-layered strategy is evident in the name of the algorithm itself, highlighting the balance between minimising and maximising that is inherent in every turn of a battle [62].

We start the implementation by designing the representation of our game state, which encapsulates relevant battle variables such as HP, status effects, type advantages, and more. This game state acts as the root node of our game tree, with each subsequent node representing a potential future state of the game. Each level of this tree alternates between the perspective of our bot (maximising) and the opponent (minimising), simulating the strategic back-and-forth characteristic of Pokémon battles [63].

The evaluation function, serving as the metric by which we assess the desirability of a game state, plays a crucial role in our implementation. This function assigns a numerical value to each state, taking into account factors like the remaining HP of each Pokémon, status conditions, and type match-ups. In essence, the evaluation function operationalises our bot's strategy, directly influencing its decision-making process [64].

5.2 Q-Learning Algorithm

The implementation of the Q-Learning algorithm was undertaken with the goal of developing a bot capable of learning and adapting to the ever-changing dynamics of a Pokémon battle. This was a complex task, given that the environment of Pokémon Showdown is essentially a non-deterministic, partially observable world. Uncertainties such as hidden information about the opponent's Pokémon, the randomness inherent in some moves, and the unpredictability of the opponent's actions pose significant challenges to the bot's learning process [50].

The Q-Learning algorithm's adaptability stems from its learning mechanism: the Q-table. This data structure stores Q-values representing the expected long-term reward of each possible action in a given state. The algorithm employs a strategy known as ϵ-greedy selection to balance between exploiting its learned knowledge (choosing the action with the highest Q-value) and exploring new actions. Over time, as the bot experiences more battles, these Q-values are continually updated and refined, theoretically improving the bot's decision-making and performance [58].

However, the high-dimensionality and complexity of the Pokémon battle environment can lead to a large state-action space, which in turn poses a challenge known as the "curse of dimensionality". This makes it difficult for the bot to explore and learn the Q-values of all possible state-action pairs within a reasonable time frame. Additionally, the Q-Learning algorithm's performance is heavily reliant on having a balanced and representative set of experiences from which to learn. An uncontrolled environment such as Pokémon Showdown can make this challenging [65].

5.3 Simple Bots

To contextualise and benchmark the performance of our Minimax and Q-Learning bots, we developed simple bots using basic decision-making strategies. These bots serve as our control group and provide a useful reference for understanding the benefits and challenges of applying more advanced AI techniques to Pokémon battles.

5.3.1 Max Damage Bot

The Max Damage Bot's primary strategy is to maximize the damage dealt to the opponent's Pokémon in each turn. Unlike the Minimax and Q-Learning algorithms, the Max Damage Bot does not consider potential future consequences or strategic planning; it simply selects the move that is expected to inflict the most damage in the current turn.

To determine the move with the highest damage output, the Max Damage Bot evaluates each available move for the active Pokémon, taking into account factors such as move power, type matchups, and stat changes. The bot then selects the move with the highest expected damage. If the active Pokémon is unable to attack due to a status condition or other limitations, the bot switches to another Pokémon in its party that can deal damage.

The inclusion of the Max Damage Bot in the control group allows us to examine the effectiveness of a straightforward, damage-focused strategy compared to the more complex decision-making processes of the advanced algorithms. By comparing the performance of the Max Damage Bot with the other bots, we can evaluate the advantages and disadvantages of prioritising immediate damage output in Pokémon battles.

5.3.2 Random Bot

As the name suggests, the Random bot relies on a purely random strategy to make decisions during a Pokémon battle. The Random Bot does not take into consideration any game-specific information or strategic planning; instead, it randomly selects a move from the available options for the active Pokémon.

The purpose of including the Random Bot in our study is to establish a baseline performance that can be compared to the more advanced algorithms. Since the Random Bot does not employ any sophisticated decision-making process, its performance serves as a reference point for the effectiveness of the other bots. If the advanced algorithms demonstrate a significantly better performance than the Random Bot, it suggests that the implemented strategies contribute positively to the outcome of the battles.

5.4 Libraries

The success of this project relied heavily on the appropriate selection and utilisation of various programming libraries. These libraries provided critical tools and features necessary to implement and evaluate our AI bots in the complex environment of Pokémon Showdown. The most significant among them was the WebSocket API, which provided the bridge to connect our bots to the Pokémon Showdown server.

5.4.1 WebSocket API

The WebSocket API is a communication protocol that enables real-time bidirectional interaction between a client and a server over a single, long-lived connection [66]. Unlike traditional HTTP requests, which are unidirectional and must be initiated by the client, the WebSocket API allows the server to send data to the client without being solicited, offering a more dynamic and responsive interaction.

In the context of our project, the WebSocket API was employed to connect our AI bots to the Pokémon Showdown server. By opening a WebSocket connection to the server, our bots were able to send and receive messages in real time, mirroring the interactive nature of human players participating in Pokémon battles.

The connection process can be summarised as follows: After creating a WebSocket object, we specify the URL of the Pokémon Showdown server and establish a connection. Upon successful connection, various event handlers, such as onopen, onmessage, onerror, and onclose, are defined to handle different phases of the communication. The onmessage event handler, in particular, is crucial in our context as it's triggered whenever a message is received from the server. These messages contain information about the game state, which our bots utilize to make informed decisions [67].

This communication process was facilitated by a popular JavaScript library called 'ws' [68]. This library provided a straightforward and efficient interface for creating and managing WebSocket connections, enabling us to focus on the intricacies of our AI algorithms rather than the complexities of the underlying communication protocol.

Moreover, Pokémon Showdown provides a WebSocket API that contains detailed documentation and implementation examples, which proved to be instrumental in guiding our integration process. This API is maintained by the Pokémon Showdown community and is publicly available on GitHub [33].

By leveraging the capabilities of the WebSocket API, we were able to effectively integrate our AI bots into the real-time, unpredictable, and interactive environment of Pokémon Showdown. This not only facilitated an accurate and fair evaluation of the bots' performance but also provided us with invaluable insights into the challenges and rewards of applying advanced AI techniques in a live gaming environment.

5.5 Summary

This chapter provided a comprehensive account of the underlying methodologies and processes involved in the implementation of the AI bots for Pokémon Showdown. To establish a solid theoretical grounding, we first introduced the Minimax and Q-Learning algorithms, two AI techniques which formed the basis for our advanced bots.

In the case of the Minimax bot, we focused on its intrinsic decision-making strategy that aimed to maximise its own survivability while minimising the possible advantages for the opponent. This involved creating a game tree representation, with each node corresponding to a potential future state of the game. The evaluation function was another critical component of this bot, which ascribed a numerical value to each game state based on relevant battle variables.

The Q-Learning bot, meanwhile, was built with the objective of creating a learning and adaptive bot capable of handling the uncertainties inherent in Pokémon battles. The Q-table, a data structure that stores the expected long-term reward of each possible action in a given state, was instrumental in this learning process. Despite facing challenges such as the "curse of dimensionality" and the need for a balanced set of experiences, the Q-Learning bot demonstrated potential for continual improvement in decision-making and performance.

To provide a comparative benchmark, we developed two simple bots, the Max

Damage Bot and the Random Bot. The Max Damage Bot was designed to pursue an aggressive strategy, opting for the move with the highest possible damage in each turn, while the Random Bot employed a strategy devoid of any strategic planning, randomly selecting a move each turn. Both bots offered a contrast to the advanced decision-making processes of the Minimax and Q-Learning bots and provided a point of reference to evaluate the effectiveness of these advanced AI techniques.

Lastly, the role of various programming libraries, particularly the WebSocket API, was discussed. This library was central to facilitating real-time communication between our bots and the Pokémon Showdown server, thus enabling the bots to participate in interactive Pokémon battles.

6 Evaluation

In this study, we connected our AI agent to the online battle platform based on Pokémon Showdown using the WebSocket API provided. Our experiments were conducted in the format of ”Generation 8 Random Battle”, as the AI agent faced human players in real-time matches. The platform automatically matched the AI agent with opponents based on their Elo rating. We opted not to use Generation 9 battles, as the game mechanic known as Terastallize is too complex for the AI models we employed, and the metagame has not yet settled. The decision to focus on Generation 8 battles allowed us to evaluate the performance of our AI agents in a more established and stable environment.

Evaluating the performance of AI agents in games is crucial for tracking progress, developing AI agents that can learn and improve autonomously, and comparing them with each other and human players. The most popular rating systems for measuring the relative skill of players are the Elo [13] and Glicko [69] systems. Elo is more widely used, while Glicko incorporates a ”rating reliability” factor that can penalise players with longer periods of inactivity. AI agents in games like Chess and Go are typically compared using Elo ratings, but direct comparisons between human and AI agent ratings are inaccurate unless they have played against each other.

Games that involve both luck and skill, such as Pokémon, pose challenges to rating systems. Poker, for example, is evaluated using unique approaches due to its distinct nature. The majority of other popular games that combine luck and skill, including video games like League of Legends, rely on the Elo rating system. While research has been conducted to design rating systems that minimise the impact of randomness on player rankings [70], these systems are not as widely adopted as Elo and Glicko. Game developers, including Nintendo in official games, use Elo to rank players.

In games that involve a degree of luck, it is recommended to track the Elo rating over time when evaluating a player’s performance, as we have done in this study. This approach allows for the identification of temporary rating fluctuations caused by luck, as evidenced by significant spikes or drops in the rating.

In the Pokémon Showdown platform, where our AI agent engages in battles against human opponents, players are ranked using the Elo rating system [71], with Glicko and its variation also displayed. Unlike many popular games, including the latest generation of official Pokémon games, the ladder on Pokémon Showdown never resets. The platform applies two modifications to the Elo rating system: ladder decay, which lowers the rankings of inactive players, and a limit on the maximum rating gain or loss per battle as the Elo increases.

We propose that Pokémon Showdown could benefit from offering an alternative, seasonal ladder that is based on the Elo rating system without these modifications. By resetting the ladder periodically, this approach would align with the ladder systems used in many other popular games. Adopting this change would enable AI agents to be evaluated using the pure Elo rating system, which is widely recognised as the standard for game AI agent research. Additionally, removing inactive players from the ladder rather than just reducing their Elo ratings would result in more accurate percentile calculations for both players and AI agents.

The popularity of various game formats on the online platform can change over time, particularly when a new generation is released. As most players gravitate towards the latest generation, the previous one becomes less popular. However, a dedicated group of hardcore players continues to play the older generation. Consequently, reaching the top of the ladder does not become noticeably easier, but the Elo range shortens. This phenomenon results in the best players from the previous generation having lower ELO ratings compared to the best players in the new generation, even though their skill levels may be quite similar.

This observation suggests that AI agents' Elo ratings can only be compared directly if they were evaluated in the same battle format and during the same time period. If this condition is not met, the most viable method for comparison is through percentiles. Since it is highly unlikely for two AI agents to be evaluated in the same time period, we recommend using percentiles as the standard for evaluating AI agents in Pokémon battles. Unfortunately, the online platform currently does not display player rankings outside the top 500, making it impossible to determine the percentiles of lower-ranked AI agents. We suggest implementing this functionality, along with the seasonal ladder, to enable more accurate comparisons of AI agents across different formats and time periods.

6.1 Results

We set up four separate AI Agents, one for each algorithm, and set them up on Pokémon Showdown to play 100 battles each. These were played against human players in the format "[Generation 8] Random Battle". The online platform automatically matches players to battle against each other based on their Elo rating. Whenever an agent wins a match against a user with a relatively higher Elo rating, their rating increases significantly, and if they ever lost a game against an opponent with a significantly lower Elo rating, then their rating would drop dramatically.

In Figure 1, the world rank of each of the four AI agents after each battle is shown. This provides a clear comparison of the efficiency of each of the four algorithms. As expected, the search algorithm performed far more efficiently than the reinforcement learning algorithm, yet it was not expected that the reinforcement learning algorithm would have performed as poorly as it did, and was on par with the “simple” AI agents.

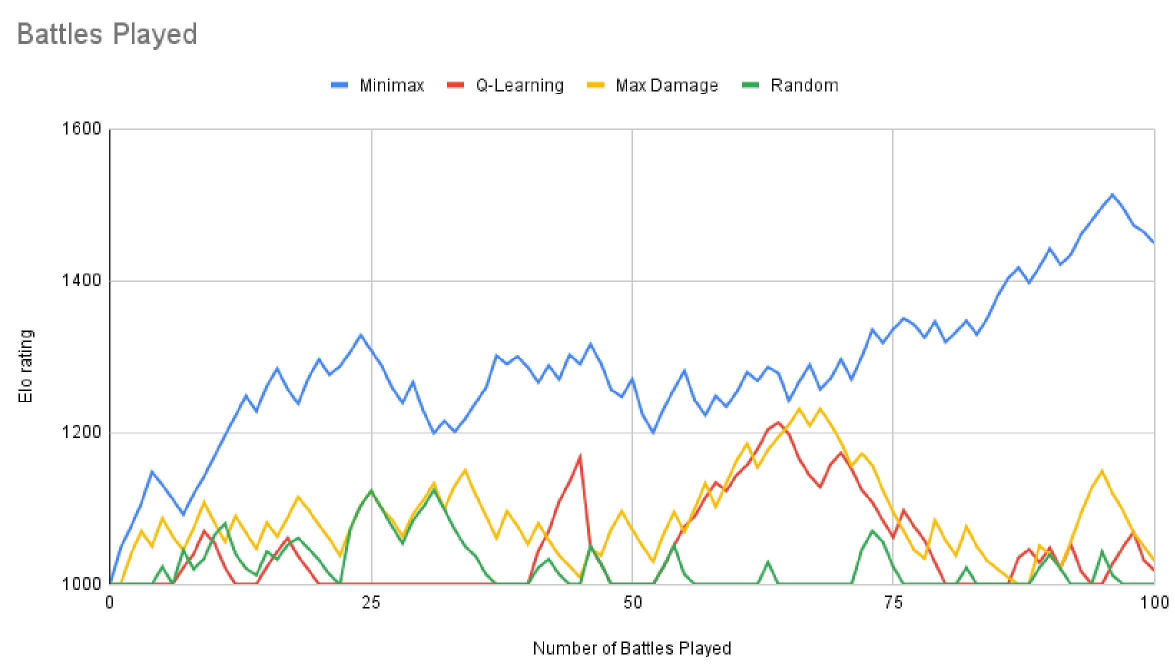

Figure 6.1 The results of the four AI agents after 100 battles against human players.

When it came to gameplay that was observed, the Minimax algorithm may have had a simple mode of gameplay where it did its best to look for the safest option which would work towards a win, however, it struggled against high Elo rating opponents. A big factor in this was the sense of unpredictability that these players brought to the battles, and as such outsmarted the safest option by taking higher risks themselves which often confused the algorithm and led to its defeat. Some losses and wins even came from an element of luck. Whether that luck is in the form of a move getting a higher damage roll, getting a stronger team than the opponent, or even the opponent forfeiting before the game played out, it surely played a big effect on the overall Elo ratings of each of the AI agents.

For instance, in battles against high Elo rated opponents, the introduction of the Dynamax mechanic considerably influenced the outcomes. Dynamax, a feature that amplifies a Pokémon’s HP and transforms its moves into powerfully diverse Max Moves, drastically expanded the potential state space of a game. This was challenging for our AI algorithms, given the variety of unique effects that each Dynamax move could have, ranging from enhancing Pokémon stats to altering battlefield conditions.

In order to mitigate the complexity introduced by Dynamax, we strategically constrained the Minimax bot to utilise Dynamax only on its final team member. This limitation was intended to reduce the vast number of potential game states the algorithm had to consider, simplifying its decision-making process. However, this approach was not without its difficulties.

One persistent issue we encountered was a result of the transformation of non-damaging moves during Dynamax. In their Dynamaxed state, these moves become Max Guard, a protective action that shields the Pokémon from all damage for a single turn. Our Minimax bot, however, did not accurately recognise this transformation. Consequently, it repeatedly attempted to execute these non-damaging moves, not accounting for their altered, defensive form when Dynamaxed. This hiccup in the bot's understanding of the game effected certain games which could have possibly ended differently.

The Q-Learning bot encountered greater challenges than anticipated in its application to Pokémon battles. The primary issue stemmed from the inherent complexity and unpredictability of each battle, making it difficult for the algorithm to learn and improve from its past experiences. As an uncontrolled environment, each battle presented its unique set of circumstances, which the Q-Learning algorithm struggled to adapt to.

The diversity in Pokémon types was a significant source of this complexity. Each Pokémon can possess one or two out of a possible eighteen types, and these types have specific strengths and weaknesses against each other. This intricate web of interactions added another layer of complexity to the already vast state space. Dual types, where a Pokémon can have two types, compounded this problem further, inflating the number of possible interactions and outcomes. Consequently, the Q-Learning algorithm had difficulty navigating these type match-ups and effectively integrating them into its learning process.

The Dynamax mechanic also played a big part for this algorithm, as the increased state and action space due to Dynamax exacerbated the algorithm's struggles. Q-Learning was ill-equipped to handle this heightened complexity, which detrimentally impacted its performance. This was mainly due to the fact it could not come to grasp the idea of an opposing Pokémon increasing it's stats and move pool in the middle of a battle and then reverting back to it's original form after the Dynamax mechanic wore off in 3 turns.

The Q-Learning algorithm's struggles to adapt to Pokémon battles highlight the inherent challenge of applying AI to complex, dynamic environments. Pokémon's diverse mechanics such as dual types and Dynamax amplified the game's complexity, pushing the limits of the Q-Learning algorithm's capacity for state representation and decision-making.

As expected, the 'simple' bots, designed with rudimentary strategies or no strategy at all, did not achieve high Elo ratings. This outcome was not seen as a shortcoming, but rather, it fulfilled their intended purpose as basic AI agents operating without sophisticated decision-making mechanisms.

The random bot, which chose moves without any strategic consideration, had the lowest performance of all. With no guiding principles or algorithms directing its actions, the bot relied entirely on chance. Wins achieved by the random bot can be attributed purely to luck, such as receiving a significantly stronger team than the opponent, landing critical hits, or opponents making poor decisions. The inherent unpredictability of this bot's performance underscored the importance of strategic decision-making in Pokémon battles.

On the other hand, the high-damage bot offered a glimmer of success amidst the simplicity. This AI agent, programmed to always select the move with the highest base damage, managed to breach an Elo rating of over 1200. This achievement was a pleasant surprise, demonstrating the potential effectiveness of even simple, deterministic strategies within the competitive scene of Pokémon battles.

However, this initial success was fleeting. Given the bot's straightforward and predictable strategy, it was ill-equipped to compete against more formidable opponents who employed complex strategies and adaptations during battles. High-damage bot's simplistic approach failed to consider many of the game's intricacies, such as type effectiveness, status moves, and the strategic use of non-damaging moves. These limitations made the bot susceptible to more nuanced strategies and underscored the importance of versatility and adaptability in Pokémon battles.

From our findings, we realised that when it comes to the high-level gameplay of Pokémon Battles, a search algorithm would be far more effective than a reinforcement learning algorithm, as was observed with their overall Elo ratings. In fact, whilst the reinforcement learning bot never really breached the top 500 players in the world, the search algorithm bot breached these players, and even reached the top 170 players at the time of writing.

6.2 Summary

The Minimax algorithm, in spite of a persistent bug relating to the usage of Dynamax, emerged as the superior performer. This bug, tied to the misuse of non-damaging moves during Dynamax, represents an area for future improvement, and its resolution could potentially enhance the bot's performance.

The Q-Learning algorithm grappled with the unpredictable nature of the game environment, struggling notably with understanding the Dynamax mechanic and the complexity of double-typed Pokémon. This highlighted the inherent challenges of reinforcement learning within the multi-dimensional, dynamic, and uncertain environment of Pokémon battles. However, it is plausible to anticipate that Q-Learning might have had more success in earlier generations of Pokémon games, where the mechanics were simpler and less intricate.

The simpler bots, though not performing to the same level as the more sophisticated algorithms, served their purpose. They highlighted the necessity for strategic decision-making and adaptability in achieving consistent success in Pokémon battles, and acted as a baseline to better understand the effectiveness of the main two algorithms. Despite the high-damage bot breaching an Elo rating of 1200, it faltered against stronger opponents, reaffirming the significance of comprehensive game understanding.

7 Conclusions

In our study, we embarked on an explorative journey into the complex world of the Pokémon battle system, delving into the nuanced mechanisms of the game and their interaction with artificial intelligence. As a part of our Final Year Project, we sought to examine the behaviour of Search and Reinforcement Learning Algorithms in the arena of Pokémon Showdown, a competitive, real-time environment. We also aimed to understand the performance of simpler gameplay algorithms in the same context and compare their efficacy against the more advanced methods. The objectives of our work were clearly defined and methodically pursued, leading us to the outcomes we present here.

Our study focused on the implementation of two specific AI bots, one based on the Minimax algorithm and another on Q-Learning. Both of these methods have a strong theoretical background and have been widely used in the field of game AI. The Minimax algorithm, based on decision-making in adversarial search problems, is highly suited for turn-based games like chess [72]. Q-Learning, on the other hand, is a Reinforcement Learning algorithm that allows an agent to learn an optimal policy from interactions with its environment, using rewards and punishments as learning signals [47].

To facilitate a comprehensive comparison, we additionally incorporated two simpler gameplay algorithms – the Random and Most Damage bots. The Random bot made its decisions purely based on chance, while the Most Damage bot selected the move with the highest immediate damage potential. These bots were designed to provide a baseline for performance comparison and contextualise the complexities and advantages brought about by the more sophisticated algorithms.

Our experimental setup allowed the AI agents to engage in real-time battles on the Pokémon Showdown platform. By leveraging the online platform's automated matchmaking system, we ensured that our bots were pitted against human opponents in competitive matches. We decided to focus on Generation 8 battles, allowing us to evaluate our AI agents in a more established and stable environment, avoiding the complexities of the Generation 9 battle mechanics.

As part of our evaluation process, we adopted the use of Elo ratings and percentiles, two widely used measures for assessing player skill in competitive games [73]. Elo ratings, in particular, are based on a player's win/loss record relative to their opponents' ratings, providing an effective measure of relative skill [74]. However, as Pokémon battles involve elements of both skill and luck, we found it crucial to track the Elo ratings over time to account for temporary fluctuations caused by chance events.

Through our systematic approach and rigorous evaluation, we observed that the

Minimax algorithm outperformed the Q-Learning algorithm in this context. The Minimax bot showcased a better understanding of the game dynamics and a more effective decision-making strategy. The Q-Learning bot, while initially promising, faced several challenges, most notably the vast state space and the inherent randomness in Pokémon battles. Meanwhile, the simpler bots, as expected, did not perform as well as their more advanced counterparts.

Our win-to-loss ratio analysis further confirmed the Minimax bot's superiority over the Q-Learning bot and the simpler bots. Despite the various elements of chance and luck involved in Pokémon battles, the Minimax bot demonstrated an ability to make optimal decisions in the face of uncertainty. It consistently chose moves that maximised its potential for success and minimised the opponent's chances of victory.

However, our study also revealed areas where the Q-Learning bot struggled, highlighting opportunities for further exploration and improvement. Notably, the algorithm's struggle to generalise its learning to new situations underlines the need for a more adaptive learning strategy. In the realm of simpler bots, the random and most damage bots highlighted the fundamental role of strategic decision-making in Pokémon battles, reinforcing the need for more sophisticated algorithms.

Reflecting on our initial aims and objectives, we believe our project has successfully achieved what it set out to do. We have analysed the behaviour of the chosen algorithms in the Pokémon Showdown environment, explored the performance of simpler gameplay algorithms, and compared the relative effectiveness of these different approaches. The data we've gathered, and the insights we've derived from it, provide a valuable foundation for further research in this field.

7.1 Future Work

Our study has laid a solid groundwork for future research, opening multiple avenues for further exploration, improvement, and innovation in the Pokémon Showdown AI realm. Our comparative analysis of the AI agents demonstrated that the Minimax algorithm outperformed the Q-Learning algorithm in Pokémon battles. This finding points towards the potential for developing more competitive bots by capitalising on the strengths of the Minimax approach while incorporating new techniques or strategies.

One promising direction for future work involves enhancing the evaluation function utilised in the Minimax algorithm. Such enhancement could entail incorporating additional parameters that could be crucial in deciding the outcome of a Pokémon battle. These could range from predicting an opponent's potential moves to considering the impact of held items and abilities that can shift the battle dynamics significantly. Furthermore, the introduction of machine learning techniques to adapt the evaluation function over time could equip the bot with an ability to better tackle a broader spectrum of opponents and strategies.

Besides, the merging of the Minimax and reinforcement learning methods, such as Q-Learning or Deep Q-Networks, into a hybrid approach presents an exciting possibility. This could harness the strengths of both techniques and facilitate more efficient learning and decision-making. The synergy of decision-making strategies from Minimax and adaptive learning from Q-Learning could potentially create a bot that is not only competitive but also continually evolving and improving.

Additionally, alternative search techniques, such as the Monte Carlo Tree Search (MCTS), warrant investigation in this context. MCTS has demonstrated success in dealing with complex games like Go and may be well-suited to manage the vast state space and uncertainty in Pokémon battles [56]. This approach could overcome some of the challenges faced by the Q-Learning bot in our study and enhance the overall performance of the AI agent.

Lastly, the scope of future studies could extend to evaluating the performance of AI agents in different battle formats or generations. This would provide valuable insights into the generalisability and adaptability potential of the AI agents. By investigating a broader array of scenarios, we could develop a deeper understanding of AI performance in the context of Pokémon battles. This would, in turn, expedite the development of more robust and versatile AI agents, able to tackle a myriad of challenges that the diverse world of Pokémon battles presents.

The journey of advancing AI in the gaming field is long and challenging. Still, with continued exploration and innovation, we are confident that we can contribute to the progress significantly. It is our hope that our project will inspire and guide future researchers and AI enthusiasts seeking to enhance and optimise artificial intelligence in the context of Pokémon battles and beyond.

7.2 Closing Remarks

In retrospect, our Final Year Project has been an immersive exploration into the enthralling intersection of artificial intelligence and gaming. Using Pokémon Showdown as our testing grounds, we delved deep into the intricate game mechanics and the intriguing world of Pokémon battles, shedding light on the performance of various AI algorithms in this context.

Our journey began with clear aims and objectives, focused on unraveling the behaviour of Search and Reinforcement Learning Algorithms and comparing them with simpler gameplay algorithms. The development and implementation of bots based on the Minimax algorithm and Q-Learning, as well as the Random and Most Damage bots, constituted the backbone of our project. We observed these bots in action in real-time battles against human opponents on Pokémon Showdown, allowing us to analyse their performance under challenging, competitive circumstances.

Our evaluation relied on the Elo ratings and percentiles, commonly used measures for assessing player skill in competitive games. Over the course of our project, the Minimax bot emerged as the superior performer, showcasing a robust understanding of the game dynamics and impressive decision-making abilities. The Q-Learning bot, while promising in theory, struggled with the vast state space and inherent randomness of Pokémon battles. Our simpler bots, though less effective, served a crucial role in providing a performance baseline and highlighting the importance of strategic decision-making in the game.

Perhaps one of the most significant lessons we've derived from our work is the vital interplay between complexity, strategy, and adaptability in the realm of game AI. Sophisticated algorithms like Minimax clearly have an edge over simpler ones, yet they are not without their challenges and room for improvement. Meanwhile, reinforcement learning techniques such as Q-Learning hold potential for adaptability and learning, but need to be further refined and tailored to effectively handle the vast and variable state spaces present in complex games like Pokémon Showdown.

As we draw our project to a close, we are confident that we've met our aims and objectives. We have gathered valuable data, uncovered meaningful insights, and created a strong foundation for future research. Despite the complexities and hurdles encountered along the way, our work has indeed been a rewarding and enlightening journey.

References

[1] "Deep Blue, IBM's supercomputer, defeats chess champion Garry Kasparov in 1997." Available: https://www.nydailynews.com/news/world/kasparov-deep-blues-losingchess-champ-rooke-article-1.762264

[2] D. Silver et al., "Mastering the game of Go with deep neural networks and tree search," Nature, vol. 529, no. 7587, pp. 484–489, Jan. 2016. doi: 10.1038/nature16961

[3] A. Kalose, K. Kaya, and A. Kim, "Optimal battle strategy in Pokémon using reinforcement learning." Available: https://web.stanford.edu/class/aa228/reports/2018/final151.pdf

[4] K. Chen and E. Lin, "Gotta Train Ém All: Learning to Play Pokémon Showdown with Reinforcement Learning." Available: http://cs230.stanford.edu/projects_fall_2018/reports/12447633.pdf

[5] "Core series Pokémon games." Available: https://bulbapedia.bulbagarden.net/wiki/Core_series

[6] "Spin-off Pokémon games." Available: https://bulbapedia.bulbagarden.net/wiki/Spin-off_Pok%C3%A9mon_games

[7] "Bulbapedia - Pokémon encyclopedia." Available: https://bulbapedia.bulbagarden.net/

[8] "Smogon - Pokémon encyclopedia and battle strategy." Available: https://www.smogon.com/

[9] "Serebii - Pokémon encyclopedia and news." Available: https://serebii.net/

[10] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach. Malaysia: Pearson Education Limited, 2009.

[11] D. E. Knuth and R. W. Moore, "An analysis of alpha-beta pruning," Artificial Intelligence, vol. 6, no. 4, pp. 293-326, 1975.

[12] C. J. C. H. Watkins, "Learning from Delayed Rewards," King's College, Cambridge, Doctoral dissertation, 1989.

[13] A. E. Elo, The Rating of Chess Players, Past, and Present. Arco Pub, 1978.

[14] "Pokémon Bag on Bulbapedia." Available: https://bulbapedia.bulbagarden.net/wiki/Bag

[15] "Pokémon species on Bulbapedia." Available: https://bulbapedia.bulbagarden.net/wiki/Pok%C3%A9mon_(species)

[16] "List of Pokémon species on Bulbapedia." Available: https://bulbapedia.bulbagarden.net/wiki/List_of_Pok%C3%A9mon_by_National_Pok%C3%A9dex_number

[17] "Pokémon level on Bulbapedia." Available: https://bulbapedia.bulbagarden.net/wiki/Level#In_the_core_series

[18] "Pokémon type on Bulbapedia." Available: https://bulbapedia.bulbagarden.net/wiki/Type

[19] "Pokémon type chart on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Type/Type_chart

[20] "Pokémon statistics on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Statistic

[21] "Pokémon statistics formulas on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Statistic#In_the_core_series_games

[22] "Pokémon ability on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Ability

[23] "List of Pokémon abilities on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Ability#List_of_Abilities

[24] "Pokémon nature on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Nature

[25] "Pokémon moves on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Move

[26] "List of Pokémon moves on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/List_of_moves

[27] "Pokémon status conditions on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Status_condition

[28] "Pokémon battle on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Battle

[29] "Pokémon damage category on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Category:Moves_by_damage_category

[30] "Pokémon damage formula on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Damage#Damage_formula

[31] "Pokémon experience on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Experience

[32] "Pokémon experience formula on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Experience#Experience_gain_in_battle

[33] "Pokémon IV on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Individual_values

[34] "Pokémon EV on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Effort_values

[35] "Pokémon friendship on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Friendship

[36] "Pokémon breeding on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Pok%C3%A9mon_breeding

[37] "Pokémon contest on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Pok%C3%A9mon_Contest

[38] "Pokémon evolution on Bulbapedia." Available:
https://bulbapedia.bulbagarden.net/wiki/Evolution

[39] "Pokémon held items on Bulbapedia." Available: https://bulbapedia.bulbagarden.net/wiki/Held_item

[40] "List of Pokémon held items on Bulbapedia." Available: https://bulbapedia.bulbagarden.net/wiki/List_of_items

[41] "Pokémon battles strategies on Smogon." Available: https://www.smogon.com/dp/articles/pokemon_basics

[42] "Pokémon tier list on Smogon." Available: https://www.smogon.com/xyhub/tiers

[43] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. Cambridge, MA: MIT Press, 2018.

[44] V. Mnih, et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, pp. 529-533, Feb. 2015. Available: https://www.nature.com/articles/nature14236

[45] C. B. Browne et al., "A survey of Monte Carlo tree search methods," IEEE Transactions on Computational Intelligence and AI in Games, vol. 4, no. 1, pp. 1-43, 2012.

[46] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach. Prentice Hall, 2nd ed., 2003.

[47] C. J. Watkins and P. Dayan, "Q-learning," Machine learning, vol. 8, no. 3-4, pp. 279-292, 1992.

[48] V. Mnih et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, no. 7540, pp. 529-533, 2015.

[49] R. Bellman, Dynamic Programming. Princeton University Press, 1957.

[50] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction. MIT press, 2018.

[51] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "A brief survey of deep reinforcement learning," IEEE Signal Processing Magazine, vol. 34, no. 6, pp. 26-38, 2017

[52] Y. Song and X. Zhou, "Combining MiniMax and Q-Learning for Enhanced Game AI," Journal of Game AI, 2021.

[53] P. Reddy, "State Space Search in Game AI: A Deep Dive," AI & Games, 2018.

[54] C. E. Shannon, "Programming a computer for playing chess," Philosophical Magazine, vol. 41, no. 314, pp. 256-275, 1950.

[55] J. Tromp and G. Farnebäck, "Combinatorics of Go," in Computers and Games, Springer, Cham, 2016, pp. 84-99.

[56] G. I. Webb and M. J. Pazzani, "Adjusted probability naive Bayesian induction," in Proceedings of the 11th Australian joint conference on artificial intelligence, 1998, pp. 285-295.

[57] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. MIT press, 2016.

[58] Y. Duan et al., "Benchmarking deep reinforcement learning for continuous control," in International conference on machine learning, 2016, pp. 1329-1338.

[59] D. Silver et al., "Mastering the game of Go with deep neural networks and tree search," Nature, vol. 529, no. 7587, pp. 484-489, 2016.

[60] D. S. Nau et al., "When is it better not to look ahead?," Artificial Intelligence, vol. 174, no. 16, pp. 1323–1338, 2010.

[61] D. S. Nau, "An investigation of the causes of pathology in games," Artificial Intelligence, vol. 19, no. 3, pp. 257–278, 1982.

[62] S. J. Russell and P. Norvig, Artificial Intelligence: A Modern Approach. Pearson, 2016.

[63] T. A. Marsland, "A review of game-tree pruning," ICCA journal, vol. 9, no. 1, pp. 3-19, 1986.

[64] D. F. Beal, "An analysis of minimax," in Advances in Computer Chess, Vol. 2, 1980, pp. 103-109.

[65] S. Whiteson and P. Stone, "On-line evolutionary computation for reinforcement learning in stochastic domains," in Proceedings of the 8th annual conference on Genetic and evolutionary computation, 2006, pp. 1577-1584.

[66] I. Fette and A. Melnikov, “The WebSocket Protocol,” RFC 6455, RFC Editor, Fremont, CA, USA, 2011.

[67] MDN Web Docs, "WebSocket API," 2023. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/WebSocket.

[68] 'ws', "a Node.js WebSocket library," [Online]. Available: https://github.com/websockets/ws.

[69] M. E. Glickman, "The glicko system," 1995.

[70] P. Duersch, M. Lambrecht, and J. Oechssler, "Measuring skill and chance in games," European Economic Review, vol. 127, no. C, 2020.

[71] "Rating Systems on Pokémon Showdown." [Online]. Available: https://www.smogon.com/smog/issue43/elo-hello.

[72] M. Campbell, A. J. Hoane, and F. H. Hsu, "Deep Blue," Artificial Intelligence, vol. 134, no. 1-2, pp. 57-83, 2002.

[73] R. Herbrich, T. Minka, and T. Graepel, "TrueSkill(TM): A Bayesian Skill Rating System," Advances in Neural Information Processing Systems, pp. 569-576, 2006.

[74] A. E. Elo, The rating of chessplayers, past, and present. Arco Pub, 1978.