PokéBOT

a Pokémon battle BOT

Artificial Intelligence
Fundamentals project,
a.y. 2022/2023

Elite Three:

- Niko Dalla Noce
- Giuseppe Lombardi
- Alessandro Ristori

Environment

- **Partially observable**, the bot doesn't know all the items, stats, moves or pokémon of the opponent.
- **Partially deterministic**, moves have an accuracy, therefore they can fail.
- **Not episodic (sequential)**, the current choice of move will affect future actions.
- **Static**, the environment doesn't change while the agent is choosing a move.
- **Discrete**, the actions an agent can take are distinct and limited.
- **Multi-agent**, battles can involve from two (our case) to four players.

Development: an evolution's tale

Just like **pokémon evolve upon reaching certain conditions**, we applied the **same approach for our bot**.

Max Base Power  **Best Damage**  **Rule-based**  **MiniMax**

Only considers the base power (not damage) of the moves

Actual damage and stats computation, implementation of switches

More accurate rules for choosing beetween a move or a switch

Tries to predict the outcome of moves

Congratulations, your bot evolved into something better, hopefully.

Cit. by us, after weeks of trial and error.

Damage and stats computation: rules over rules

The **damage dealt by a move** is computed by means of the following formula.

$$Damage = \left(\frac{\left(\frac{2 \times Level}{5} + 2 \right) \times Power \times A/D}{50} + 2 \right) \times Targets \times PB \times Weather \times Critical \times random \times STAB \times Type \times Burn \times other$$

Attacker and defender stats

[0.85, 1]

- The **more accurate is the damage**, the **better the choice of moves** from the bot.
- The **computation is made up of many rules** that can be explained in FOL.

E.g: if a pokémon is holding the «air baloon» item, then it's immune to ground-type moves.

$$\forall x\ Pokemon(x) \wedge \forall y\ Move(y) \wedge Holds(x, Air\ Balloon) \wedge MoveType(y, Ground) \rightarrow Immune(x, y)$$

The same process is **also applied for the stats computation**.

Matchup: a «weak» dominant strategy in action

Damage multiplier table

*We consider the **type of the pokémon and their moves**.*

Matchup value [-8, 8]

Opponent's pokémon

Bot's pokémon

VS	ACT		UNK	UNK	UNK	FAINTED
ACT	-3	0	0	0	0	/
	2.5	0.5	0	0	0	/
	0	1	0	0	0	/
	1	0	0	0	0	/
FAINTED	/	/	/	/	/	/
FAINTED	/	/	/	/	/	/

*With a bad matchup **we need to switch to the pokémon with highest value**, assuming the **opponent's pokémon won't change**.*

Putting everything together: a rule-based player

The first «enhanced» playstyle for the bot was **built by using all what we've seen until now** plus some little improvements after asking for help to a friend who acted as a field-expert:

- **Healing moves** are executed only if the expected damage from the opponent is less than the healed hp.
- **Priority moves** are used if their damage is enough to defeat the opponent's active pokémon.
- **Boost moves** are used if the opponent's pokémon is not a threat (this implies a good matchup for the bot).
- **Protective moves** are used, sometimes, when we don't know what move to expect from the opponent's pokémon.
- **Switches** are executed under specific conditions (e.g.: a matchup less than -1 or when the active pokémon has been suffering from toxic for many turns).

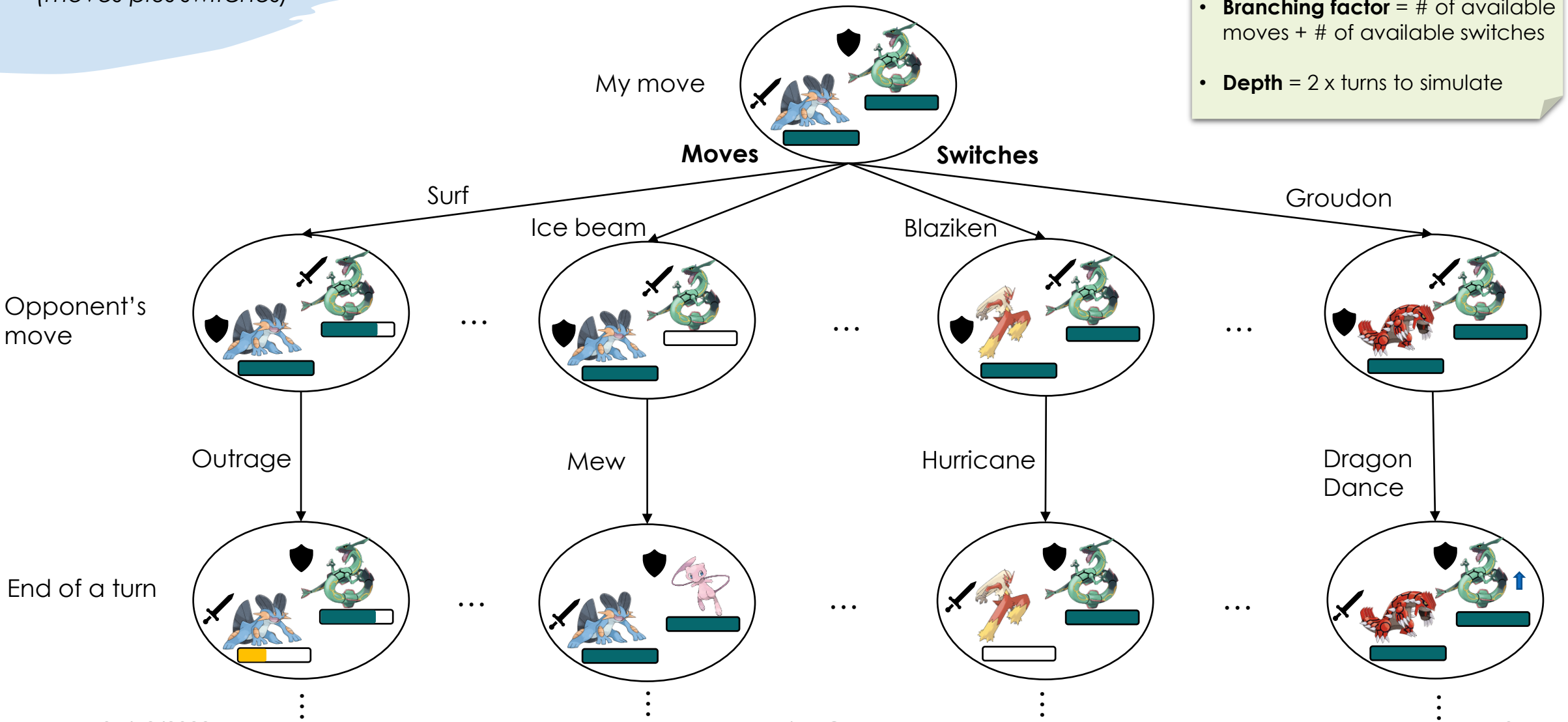
The bot started acting pretty well, but we need to predict the opponent's next moves, just like a human player.

minimax

MiniMax with α-β pruning

(moves plus switches)

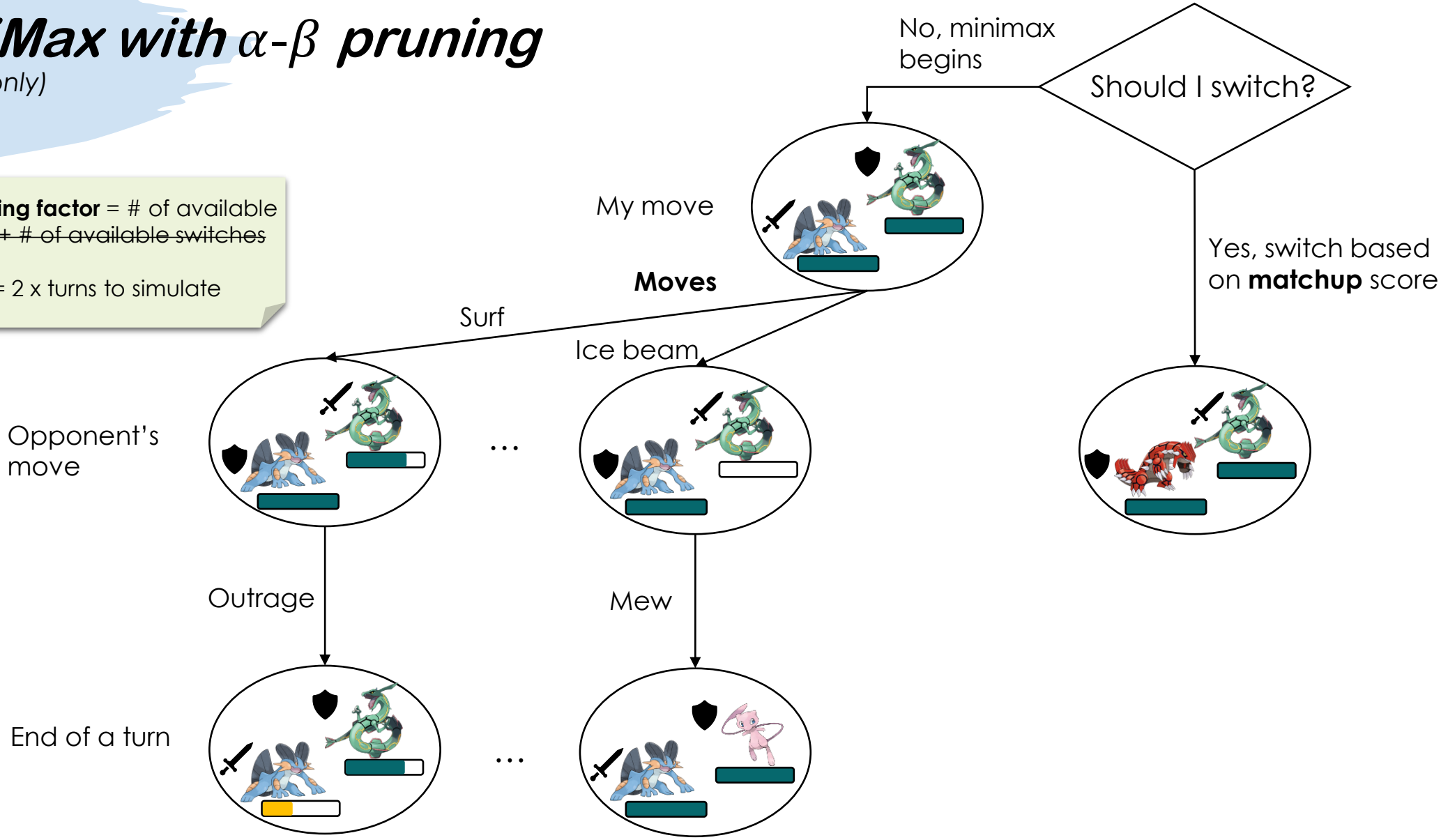
- **Branching factor** = # of available moves + # of available switches
- **Depth** = 2 x turns to simulate



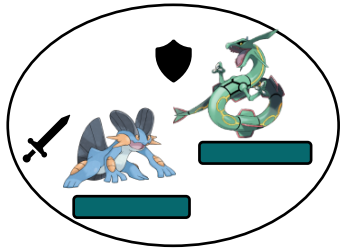
MiniMax with α-β pruning

(moves only)

- **Branching factor** = # of available moves ~~+ # of available switches~~
- **Depth** = 2 x turns to simulate



A minimax node



In a battle, the opponent's team, Pokémon statistics, abilities and moves can not be known a priori.

Relevant **informations** to **store:**

- **Our team:**  …
- **Opponent's team:**  …
- **Weather conditions:**


Sandstorm

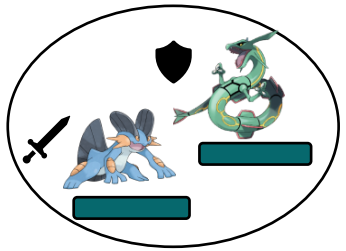

Harsh sunlight


Rain

…

- **Pokémon moves:** surf, outrage, ice beam, earthquake, etc
- **Pokémon statistics:** hp, attack, special attack, defence, special defence, speed
- **Pokémon boosts:** how much the base statistics are increased

A minimax node



In a battle, the opponent's team, Pokémon statistics, abilities and moves can not be known a priori.

Relevant **actions** to **simulate**:

- **Damage:** how many hp the opponent Pokémon lost after my attack.
- **Recoil:** some moves substact hp also to our Pokémon due to a recoil.
- **Drain:** there are moves that damage the opponent and in the meanwhile can restore some health point of our Pokémon.
- **Weather conditions:** the damage of a move and some statistic of a Pokémon are influenced by the weather conditions.
- **Boost changes:** it is important to keep trace of this information in order to know if an opponent's move with some boosts can deafeat our Pokémon and vice versa.
- **Opponent's team:** the objective is to simulate the status of the opponent's team after the end every turn.
- **Our team:** simulate the status of our team.

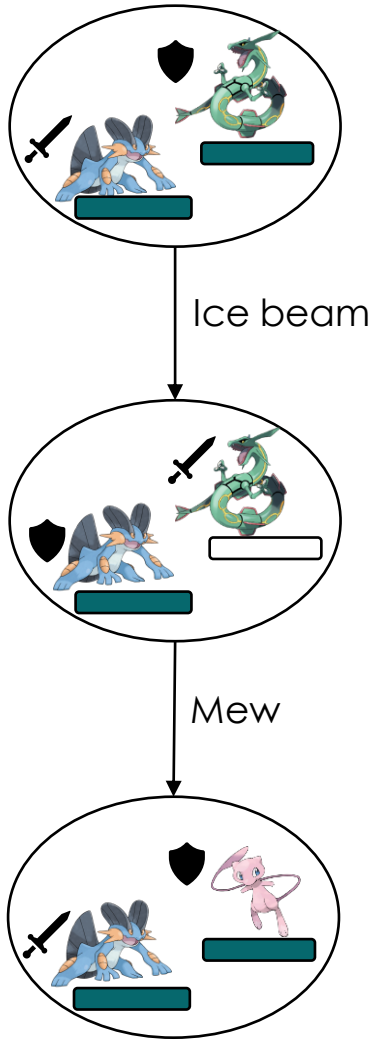

heuristic

Evaluation Function

- First, it was just a **simple function:**

$$\overline{hp}(\cdot) = \frac{hp(\cdot)}{hp_{total}\,(\cdot)}$$

$$f(node) = \overline{hp}(active_{MAX}) - \overline{hp}(active_{MIN})$$



Evaluation Function

- **Linear weighted sum of features:**

$$f(node) = w_1 \cdot \overline{hp}(team_{MAX}) + w_2 \cdot alive(team_{MAX}) - w_3 \cdot \overline{hp}(team_{MIN}) - w_4 \cdot alive(team_{MIN}) - p \cdot depth(node)$$

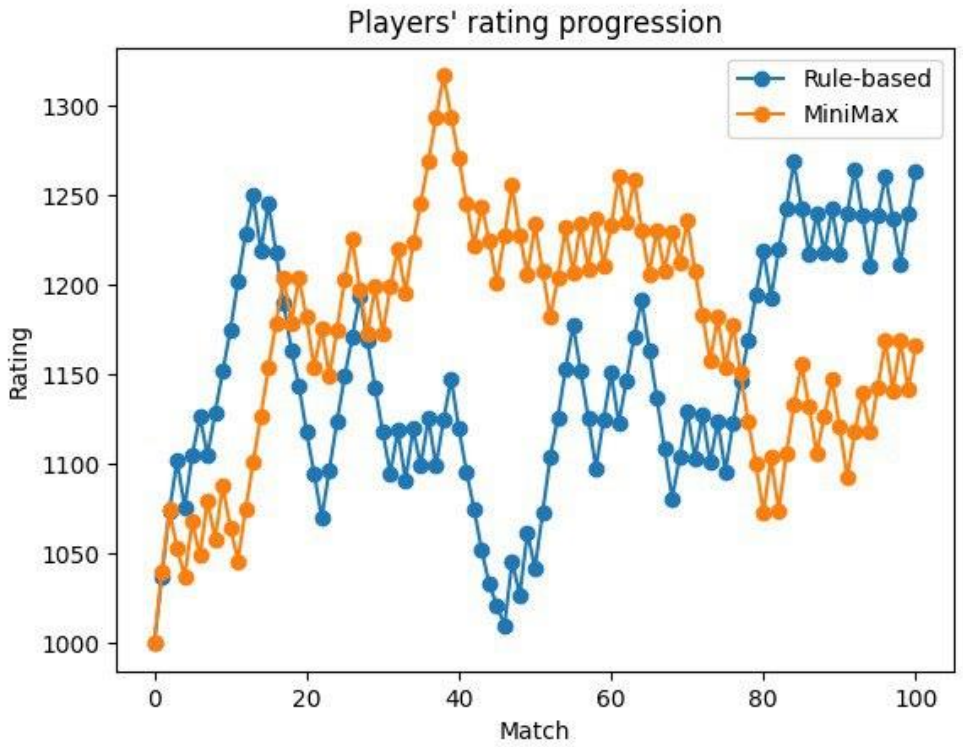
  - Knowledge on Pokemon teams
  - Penalty term on depth → enhance exploration
  - Random search on weights
  - Assessed on the percentage of wins achieved against a baseline player

results

Results

Player	MaxBasePower	BestDamage	RuleBased	MiniMax
MaxBasePower	—	0.104	0.084	0.114
BestDamage	0.896	—	0.398	0.452
RuleBased	0.916	0.602	—	0.571
MiniMax	0.886	0.548	0.429	—

- **1000 simulated matches** between our bot's different playstyles to assess the strength of each one.
- **RuleBased** and **MiniMax** are the strongest players as expected.



- RuledBased and MiniMax played autonomously ladder (competitive) matches **against humans**.
- The ELO score shows how robust the bot is **against a human opponent.**

Future work and improvements

- **More rules** for the **damage and stats computation**.
- Implementation of all the **changes coming from the new generation**.
- **Improvements** on the **minimax player** by taking into account the **accuracy and secondary effects** of each move.
- Implementation of a **reinforcement learning player**.

Do you wanna be the very best?

The **bot is up for the challenge**, **follow the instructions** on our github repo to **test your pokémon battle ability**.

https://github.com/nikodallanoce/PokeBOT

Are you ready for a little demo?

References

Main paper about pokémon battles:

- S. Lee and J. Togelius, "Showdown AI competition," 2017 IEEE Conference on Computational Intelligence and Games (CIG), 2017, pp. 191-198, doi: 10.1109/CIG.2017.8080435. Showdown AI competition | IEEE Conference Publication | IEEE Xplore

Some **cool implementations**:

- https://github.com/pmariglia/showdown
- RemptonGames/Pokemon-Showdown-Agent (github.com)

Main library:

- https://github.com/hsahovic/poke-env