Genetic Algorithm to Create Pokemon Teams Based on Typing

Sebastian Garcia

December 2023

1 Abstract

Role-Playing Games, or RPGs, are some of the most popular and played genre of games. One such game, Pokemon, is the most profitable and iconic franchise. Their main line of games are a series of RPGs where the player, or trainer, travels the map catching Pokemon and battling trainers with the player's own team of Pokemon. With a vast amount of Pokemon to choose for the player's team as well as many opposing trainers the player must fight, choosing a strong and balanced team to play the game with can be a difficult task. This paper proposes a traditional genetic algorithm to determine a strong team to use against a list of trainers and their Pokemon in regards to the type combinations of the player's team and Pokemon the player will be facing. The results show this algorithm can produce good teams as well as narrow the pool of Pokemon to choose from considerably as well as the moves those Pokemon should be using. While the algorithm does not produce a single best team in the set number of generations, it does consistently produce good teams. This may be favorable, due to certain players not wanting to use the best Pokemon, choosing to use Pokemon they prefer more, and no truly best team in Pokemon as there a lot of other factors that go into team selection other than typing.

2 Introduction

Pokemon is a Role-Playing Game where the player uses a team of six Pokemon to battle trainers and their Pokemon throughout the game, inevitably leading to the final set of battles against the Elite Four, four of the strongest trainers in the game, and finally, the Champion, who is the final and strongest trainer in the game. A battle consists of one member from each trainer's team battling in a turn based battle where one move is used by each Pokemon per turn. These moves deal damage to the opposing Pokemon until one of the Pokemon's health points reaches zero.

Each individual Pokemon consists of many changeable features, such as the moves that Pokemon knows, only being able to know up to four moves at a time, and many unchangeable features, such as the Pokemon's typing, which can be up to two types, and moves it can learn, which is determined by the creators of the game. In regards to typing, as of now there are 18 types, each of which has an interaction with another type, either being neutral, super-effective, resisted, or no-effect. When a Pokemon attacks with a move, the moves type is checked against the other Pokemon's typing to determine the interaction of that type move using a type chart.

The algorithm used is a traditional generational genetic algorithm. The population consists of teams, where each team consists of its fitness and six Pokemon. Each Pokemon consists of its fitness, the name of the Pokemon, its typing, and the four types for its moves. It employs two crossover operators,

single-point crossover, and a variation on uniform crossover. It employs two mutation operators, one replaces a Pokemon in a team, and the other replaces a move of a Pokemon. Both the rates for each crossover and mutation operation change during the run of the algorithm. For selection, a binary tournament is used. The fitness of a Pokemon is determined by how well the types of its attack do against the typing of the opponent Pokemon plus how well the types of the opponent's attacks do against the player's Pokemon's types. The fitness of a team is the sum of the fitness of the Pokemon on that team with the goal to maximize the fitness.

The motivation of this paper is to show a genetic algorithm can be used to find optimal teams to use against a set of Pokemon. The set of Pokemon chosen for this came from the fifth generation of Pokemon, which correlates to the games Pokemon Black and White. The reason for this choice was the large list of Pokemon available, around 70. Using Serebii.net, a site containing data on Pokemon, the available Pokemon were collected along with filtering all reasonable types for that Pokemon to have for its moves. Status moves, moves which do not do damage, were not considered due to the complexity they introduce into battles along with abilities, held items, etc. Only typing was considered for this algorithm. This algorithm was able to provide increases in fitness resulting in better teams to take on the set of opponent Pokemon.

3 Related Works

The paper "An Approach for Team Composition in League of Legends using Genetic Algorithm," 2019, L. M. Costa, A. C. C. Souza and F. C. M. Souza was used as a basis for this paper.

The paper by Costa et al. focused on a similar concept but applied to the game League of Legends. They used a genetic algorithm, but rather than compare the team to another team, they used the stats of characters to determine the fitness of the team from three fitness functions. This paper also uses a genetic algorithm, but employs a different fitness function. This paper also uses single point crossover and a replace mutation, but adds a new crossover operator and mutation operator. On first testing of this algorithm, the probabilities, population size, and number of generation they stated in their work were used. These were tweaked to improve the algorithm, in particular, I found greater success when increasing the mutation probability and number of generations.

4 Methodology

4.1 Data Collection

Using Serebii.net, the set of available Pokemon were chosen, and from each Pokemon, their moves and typing was taken and saved as strings in arrays. Thus, each Pokemon was represented by an array whose format followed *[Pokemon's name, [Pokemon's types], [Type of Moves]]*. One example of this format would be *["MANDIBUZZ", ["DARK", "FLYING"], ["NORMAL", "FLYING", "DARK", "GROUND", "BUG", "ROCK"]*. The set of all viable Pokemon was another array consisting of the arrays of Pokemon following the format stated previously. For the type chart, a 2-D array was used. The 2-D array works by converting the string of the types to an int, which is used as the indices of the 2-D array. In each cell of the 2-D array is an int representing the effectiveness of that type combination. For example, the type Normal correlates to the int 0. So if a Normal type move was used on a normal type Pokemon, you would look at the cell [0][0] in the 2-D array. Within the cell would be the int 1 representing neutral effectiveness of Normal on Normal. All this data was collected

by hand and filtered by hand.

The set of Pokemon to be faced by the player's team was pulled from the Elite Four of the fifth generation of Pokemon, the games Pokemon Black and White, using Serebii.net. Each of the Elite Four used four Pokemon and the Champion used six. These Pokemon were compiled with their typing and typing of their moves, excluding status moves, the same way individuals were compiled for the player's team and into one array.

4.2 Genetic Algorithm Representation, Parameters, and Fitness

For the genetic algorithm, the number of generations was set to 1500, and the population size was set to 30. The initial population is create by creating 30 teams. Each team is created by randomly selecting six Pokemon from the list of valid Pokemon, thus it consists of six arrays and an int for the fitness. An individual Pokemon consists of an array containing the fitness represented by an int, initialized to 0, and the array containing the name of the Pokemon, the typing of the Pokemon in an array, and the moves of the Pokemon also in an array. The format of an individual would be represented by the format *[fitness int, [Pokemon pulled from array of viable Pokemon]]*. The fitness is updated once the Pokemon is created by running a match up function which compares the individual Pokemon against the set of opponents and assigns the fitness to the index labeled *fitness int* in the format given above.

The fitness for a team is calculated by summing the fitness of each Pokemon on that team. The

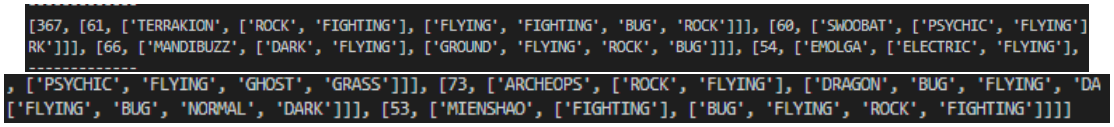

Figure 1: An example of the best team from a run. The first int in the top image is the overall fitness of the team. The int in front of each Pokemon is that Pokemon's fitness, then followed by that Pokemon's name, typing, and types of their moves. The image is split to fit on the page.

fitness of a Pokemon is calculated by comparing that Pokemon to each Pokemon on the set of Pokemon to be faced and is initialized to 0. Points are assigned to the fitness based on the effectiveness of the moves. Two points are added if the move is super-effective, and one point is taken away if the move is resisted. The reasoning behind this point allocation is that it is more important to have a super-effective move than one of the moves be resisted since only one move is used per turn in a battle. Another point is added if the move is the same type as the Pokemon it belongs to as in the game Pokemon, bonus damage is done if the move is of the same type as the Pokemon. The same is done for the opponents attacks on your Pokemon and that sum is subtracted from your Pokemon's fitness. A penalty is applied to the fitness if the team has two or more of the same Pokemon, subtracting the fitness of those Pokemon from the team's fitness

4.3 Crossover

Crossover occurs with probability 1.0 to create children equal to the size of the population minus one. Elitism is used so the last individual added to the population is the best individual from the previous

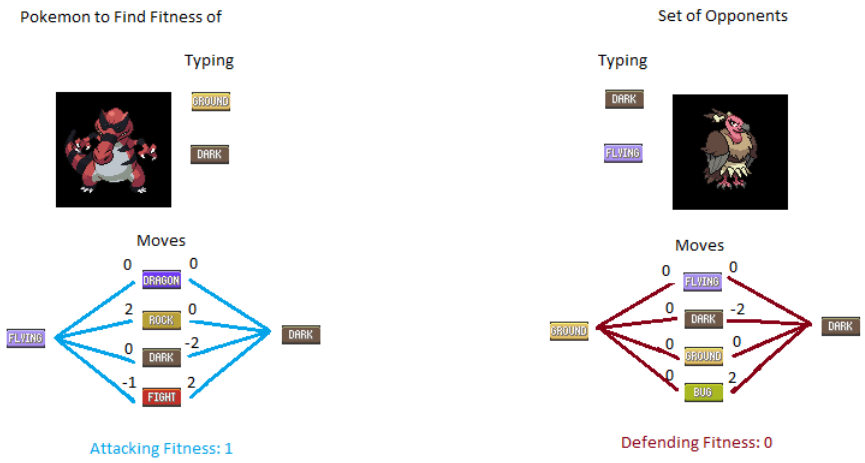

Figure 2: An example of how the fitness of Pokemon were calculated. The Pokemon on the right is the Pokemon to find the fitness of. It's moves are compared to the typing of the opposing Pokemon and vice versa. The move's effectiveness points are summed to calculate that Pokemon's fitness for both Pokemon. The overall fitness is calculated by subtracting the opposing Pokemon's fitness from your Pokemon. This can be interpreted as being how good your Pokemon is subtracted by how good the opposing Pokemon is on you. Of note, the match up Dark on Dark is -2 since the move is resisted meaning -1 and since it is the Pokemon is also Dark type, another -1 penalty is added.

generation. Single point crossover occurs with probability 0.7. The modified uniform crossover occurs with probability 0.3. This modified crossover works by adding to the child all Pokemon shared between the two parents and chooses the remaining Pokemon to fill the team from randomly selecting a Pokemon from the parents. Selection is done via binary tournament selecting both parents for crossover.

4.4 Mutation

Mutation occurs with probability 0.7. If mutation probability succeeds, it mutates one member in the population. This seemed reasonable as the population size is small and the mutation rate is high. During the first three fourths of generations, the mutation that replaces a Pokemon occurs with probability 0.8 while the mutation that replaces a Pokemon's move occurs with probability 0.2. These change to 0.1 and 0.9 respectively after three fourths of the generations have been completed. The reasoning is that replacing a Pokemon is more exploratory and causes big changes in the fitness, so this works better at the beginning of the run to determine the best Pokemon. Afterwards, using the mutation changing the moves causes small changes in the fitness, so this would work better towards the end of the run to determine the better moves to have on the Pokemon.

Figure 3: An example of the modified uniform-like crossover operation. Similar Pokemon between the two parents are kept in the child while the remaining slots to be filled are chosen by random from either parent. The top parent in the figure is actually one of the maximum fitness found individual throughout the 30 runs, with a fitness of 399. The bottom parent was made from other Pokemon found in the best teams throughout the runs.

4.5 Experiment

30 runs of the algorithm were done, collecting the best fitness achieved and the team associated with that fitness. The frequency of each Pokemon and types of moves for those Pokemon were collected. Each run took around one minute on an AMD Ryzen 7 8-Core Processor in VS Code where the genetic algorithm was written from scratch in Python.

5 Analysis

The average fitness of the first generation over the 30 runs was 205.9 while the average fitness over the first 30 runs of the best individuals from each run was 383.267 with a maximum of 399. Totaling the Pokemon found over these 30 runs, there were clear Pokemon that were favored for fighting this set of trainers, the Elite Four and Champion from Pokemon Black and White. The maximum fitness

RUN (xprc	BEST FIT	MEMBER	MOVE	MOVE	MOVE	MOVE	MEMBER	MOVE	MOVE	MOVE	MOVE	MEMBER	MOVE	MOVE	MOVE	MOVE
1	387	archeops	flying	bug	fighting	rock	swanna	water	flying	normal	ice	swoobat	psychic	flying	grass	ghost
2	373	emboar	fire	dark	ground	steel	sigilyph	flying	ice	electric	grass	volcarona	fire	flying	grass	normal
3	368	chandelu	fire	grass	ghost	normal	swoobat	psychic	flying	ghost	normal	volcarona	flying	bug	normal	fire
4	395	archeops	flying	ground	dark	fighting	mandibu	dark	flying	rock	ground	swanna	water	flying	normal	ice
5	387	swanna	water	flying	normal	ice	emboar	ground	fire	electric	fighting	sigilyph	flying	psychic	electric	ice
6	378	swanna	water	flying	normal	ice	simipour	flying	dark	fighting	ghost	emboar	ground	steel	dark	fire
7	372	darmanit	fire	fighting	bug	dark	swanna	water	flying	normal	ice	cobalion	fighting	bug	rock	flying
8	383	archeops	dragon	fighting	flying	bug	swanna	water	flying	normal	ice	sigilyph	flying	steel	ghost	ice
9	375	swanna	water	flying	normal	ice	simisear	rock	ghost	flying	fire	emolga	electric	normal	flying	bug
10	396	sigilyph	flying	ghost	ice	steel	archeops	flying	dark	dragon	bug	mandibu	flying	dark	bug	rock
11	385	archeops	flying	ground	fighting	dark	simisear	ghost	fire	dark	flying	swanna	water	flying	normal	ice
12	378	mandibu	flying	dark	rock	ground	emboar	normal	fire	ground	rock	swanna	water	flying	normal	ice
13	392	simisear	fire	flying	fighting	ghost	archeops	flying	fighting	ground	dark	emboar	fire	steel	fighting	rock
14	385	gyarados	normal	water	ice	rock	emboar	fighting	electric	fire	ground	swanna	water	flying	normal	ice
15	373	swoobat	grass	ghost	psychic	flying	cobalion	rock	flying	fighting	electric	archeops	flying	rock	dark	ground
16	381	golurk	normal	ghost	ground	flying	swoobat	psychic	flying	ghost	normal	mandibu	flying	rock	bug	ground
17	386	swanna	water	flying	normal	ice	archeops	flying	rock	bug	fighting	swoobat	psychic	flying	grass	ghost
18	387	golurk	flying	ground	dark	ghost	swanna	water	flying	normal	ice	swoobat	psychic	flying	grass	ghost
19	391	terrakion	normal	fighting	ground	flying	chandelu	fire	ghost	normal	psychic	swanna	water	flying	normal	ice
20	389	chandelu	fire	ghost	normal	psychic	swoobat	ghost	normal	psychic	flying	emboar	fire	fighting	ground	dark
21	399	sigilyph	flying	ice	ghost	electric	terrakion	normal	fighting	bug	flying	archeops	flying	rock	dark	ground
22	369	gyarados	rock	water	ground	dark	golurk	ghost	ground	fighting	flying	swanna	water	flying	normal	ice
23	378	swanna	water	flying	normal	ice	swoobat	psychic	flying	normal	grass	mandibu	flying	ground	bug	dark
24	399	terrakion	bug	ground	flying	fighting	archeops	rock	fighting	flying	bug	simisear	ghost	flying	rock	fire
25	379	mandibu	rock	ground	flying	bug	archeops	dark	bug	flying	ground	terrakion	normal	rock	bug	flying
26	385	emboar	fire	fighting	dark	ground	archeops	flying	dragon	normal	ground	swoobat	flying	ghost	psychic	grass
27	370	gyarados	water	ice	fighting	rock	sigilyph	flying	ghost	ice	psychic	archeops	normal	flying	rock	dragon
28	390	archeops	ground	rock	flying	fighting	simisear	fire	dark	flying	ghost	chandelu	fire	ghost	normal	poison
29	398	terrakion	flying	ground	fighting	bug	archeops	dark	bug	flying	fighting	swanna	water	flying	normal	ice
30	370	terrakion	fighting	rock	flying	poison	simisear	rock	ghost	fire	flying	mandibu	flying	normal	bug	ground

MEMBER	MOVE	MOVE	MOVE	MOVE	MEMBER	MOVE	MOVE	MOVE	MOVE	MEMBER	MOVE	MOVE	MOVE	MOVE	BEST FIT AT GEN = 0
mandibu	flying	ground	bug	rock	emboar	fire	normal	ground	steel	golurk	flying	ghost	ground	dark	160
archeops	flying	dragon	fighting	normal	swanna	water	flying	normal	ice	terrakion	normal	fighting	ground	flying	204
terrakion	bug	normal	flying	fighting	swanna	water	flying	normal	ice	archeops	flying	dark	dragon	fighting	217
emboar	fire	rock	electric	normal	swoobat	psychic	ghost	grass	flying	volcarona	fire	flying	bug	psychic	175
mandibu	normal	flying	dark	bug	archeops	normal	dark	flying	bug	simisear	ghost	fire	flying	rock	220
archeops	flying	rock	fighting	dragon	terrakion	bug	flying	fighting	rock	mandibu	flying	ground	bug	dark	182
mandibu	flying	dark	ground	rock	archeops	bug	ground	dark	flying	simisear	fire	dark	flying	fighting	186
emboar	fire	normal	steel	electric	terrakion	rock	ground	fighting	flying	braviary	flying	fighting	bug	ghost	207
emboar	normal	electric	fire	dark	chandelu	psychic	normal	fire	ghost	mandibu	flying	ground	bug	rock	201
terrakion	fighting	bug	ground	flying	swanna	water	flying	normal	ice	emolga	electric	dark	flying	bug	206
braviary	flying	fighting	ghost	bug	emolga	electric	dark	flying	bug	chandelu	fire	psychic	normal	ghost	181
archeops	flying	normal	rock	fighting	emolga	electric	dark	flying	bug	swoobat	grass	ghost	normal	flying	200
braviary	flying	bug	rock	ghost	volcarona	psychic	grass	flying	fire	swanna	water	flying	normal	ice	193
archeops	normal	flying	fighting	dark	mandibu	flying	dark	ground	bug	chandelu	fire	ghost	grass	psychic	201
swanna	water	flying	normal	ice	chandelu	fire	ghost	grass	psychic	mandibu	normal	dark	ground	flying	235
archeops	flying	fighting	ground	bug	swanna	water	flying	normal	ice	chandelu	fire	ghost	psychic	grass	235
emboar	rock	dark	ground	electric	mandibu	flying	dark	rock	ground	terrakion	bug	ground	flying	normal	177
emolga	electric	dark	flying	bug	terrakion	fighting	rock	bug	flying	archeops	flying	dark	fighting	bug	219
emboar	dark	electric	fire	fighting	sigilyph	flying	ghost	ice	grass	archeops	normal	flying	dark	ground	240
swanna	water	flying	normal	ice	emolga	electric	dark	flying	bug	archeops	flying	dark	ground	rock	220
swoobat	psychic	flying	ghost	grass	emboar	fire	rock	dark	ground	swanna	water	flying	normal	ice	228
chandelu	fire	ghost	psychic	normal	emboar	rock	steel	dark	electric	mandibu	flying	ground	bug	dark	200
emboar	electric	ground	fighting	fire	archeops	normal	dark	dragon	flying	golurk	rock	ghost	flying	fighting	152
sigilyph	flying	electric	psychic	ghost	swanna	water	flying	normal	ice	emboar	dark	fire	normal	rock	174
swanna	water	flying	normal	ice	sigilyph	flying	electric	ghost	normal	braviary	flying	fighting	bug	ghost	236
chandelu	poison	fire	ghost	grass	mandibu	ground	bug	flying	rock	swanna	water	flying	normal	ice	212
terrakion	normal	bug	flying	fighting	chandelu	fire	ghost	grass	psychic	swanna	water	flying	normal	ice	305
emboar	dark	electric	fighting	fire	swanna	water	flying	normal	ice	terrakion	bug	ground	flying	rock	165
emboar	electric	steel	fighting	dark	mandibu	ground	dark	rock	flying	chandelu	grass	psychic	ghost	fire	210
volcarona	psychic	flying	bug	fire	swanna	water	flying	normal	ice	emboar	fighting	electric	fire	steel	236

Figure 4: The members of each team across the 30 runs. Each member has the four moves they have to their immediate right. The second column is the best fitness achieved in the population over the entire run. The last column is the best fitness of the population at generation 0. A run best team is read left to right across both tables. The information could not fit horizontally in one table.

Archeops		Emboar		Chandelure		Swanna		Darmanitan		Sigilyph		Mandibuzz		Simisear
flying	27	fire	18	fire	13	water	30	fire	1	flying	9	flying	18	fire
dark	16	dark	12	ghost	13	flying	30	fighting	1	ghost	7	ground	16	flying
fighting	16	electric	12	psychic	10	normal	30	bug	1	ice	7	dark	12	ghost
ground	12	ground	11	grass	7	ice	30	dark	1	electric	5	bug	12	rock
bug	11	fighting	10	normal	7			normal		psychic	3	rock	11	dark
rock	10	steel	8	poison	2			rock		steel	2	normal	3	fighting
normal	8	rock	7					steel		grass	2			normal
dragon	8	normal	6							normal	1			
		poison												

Gyarados		Swoobat		Golurk		Terrakion		Simipour		Cobalion		Volcarona		Emolga		Braviary	
water	3	flying	12	ghost	5	flying	15	ghost	1	fighting	2	fire	5	electric	6	flying	4
rock	3	psychic	11	flying	5	fighting	12	dark	1	rock	2	flying	5	flying	6	bug	4
ice	2	ghost	11	ground	4	bug	11	flying	1	flying	2	bug	3	bug	6	ghost	4
normal	1	grass	9	fighting	2	ground	8	fighting	1	bug	1	psychic	3	dark	5	fighting	3
dark	1	normal	5	dark	2	normal	7	normal		electric	1	normal	2	normal	1	rock	1
fighting	1			normal	1	rock	6	water		normal		grass	2			normal	
ground	1			rock	1	poison	1	rock		steel							
dragon				steel						poison							

Figure 5: Part of set of Pokemon and the occurrences of their moves in the first 30 runs.

of the population increases logarithmically throughout the run

5.1 Specific Best Pokemon Found

Out of the 75 total Pokemon to choose from, only 17 ever appeared on the best team of that run. Within these 17, some were favored more than others. The Pokemon *Swanna* appeared the most in 30 of the 30 teams. *Archeops* appeared in 27 of the teams, and *Emboar* appeared in 21. For the moves of these Pokemon, *Swanna* only had four moves to choose from, so it always had those four moves. *Archeops* had 8 moves it could choose from. Of those 8, Flying appeared 27 out of the 27 times *Archeops* was used, followed by Dark and Fighting appearing 16 times, while moves like Dragon and Normal only appeared 8 times. This can be interpreted as *Archeops* with the moves of the types Flying, Dark, Fighting, and Ground is an optimal Pokemon to bring to face this Elite Four and Champion. The reason for this is due to three of the Elite Four members using the types Fighting, Psychic, and Dark (Elite Four members are known to exclusively use one type). Flying is good against Fighting, Dark against Psychic, and Fighting against Dark. Ground is a good move that is only resisted by Grass and Bug and since no Grass Pokemon exist among the opponent's Pokemon and only three Bug types are on the Champions team, this is why bringing these moves on *Archeops* are better than other choices.

It is possible that none of the maximum fitness values for the teams found during the 30 runs are the global optimum, but given more runs, the algorithm may find as it always seems to get really close, but only further testing will answer this. It is also important to note this algorithm will most likely never "The Best Pokemon Team" to fight against the set of trainers given. It only works off of typing of the Pokemon and typing of their moves. There is much more that goes into a Pokemon battle than typing. Abilities, stats of Pokemon, status moves, power of moves are a few important ones, however given time and complexity constraints, these were omitted from the fitness function, but may be an area of future work.

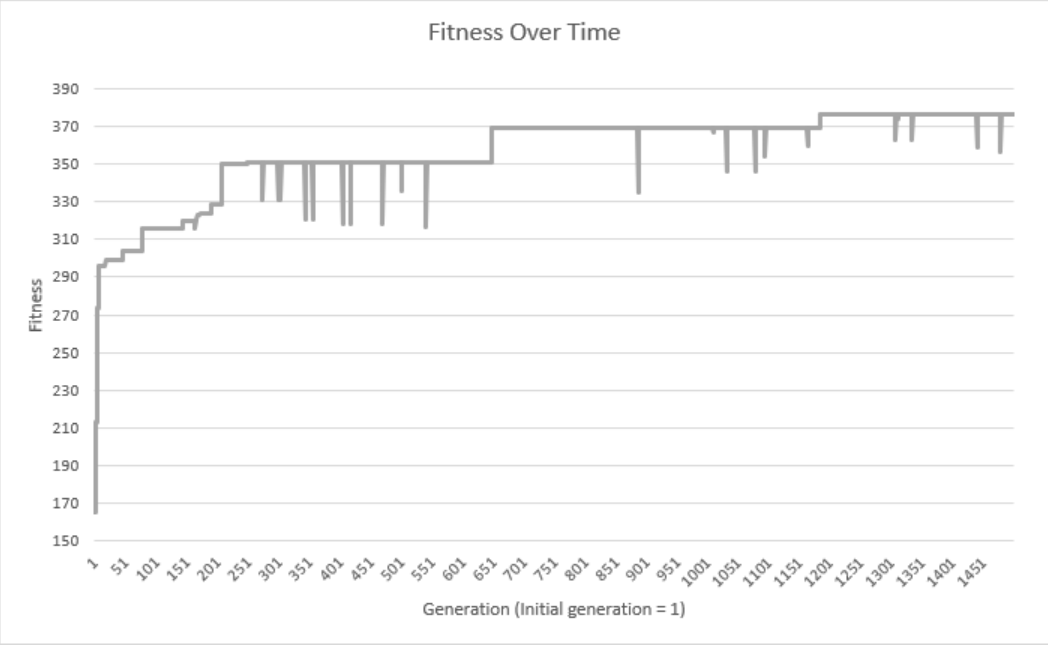

Figure 6: The maximum fitness of the population throughout the run.

6 Future Work

I would like to look into improving the fitness function to include more information about Pokemon to make better teams. Making the fitness function multi-objective may prove beneficial in finding better teams, accounting for base stats of the Pokemon, power of the moves, and usefulness of the abilities of that Pokemon. I would also like to look into using other approaches than a genetic algorithm. I considered using evolutionary strategies, but since this is a non-continuous, discrete search space, I decided to avoid it, especially due to time constraints. A steady-state genetic algorithm may also be beneficial to look into since it could preserve better individuals better than the genetic algorithm. Also, looking into better mutation and crossover operations may prove beneficial, especially intelligent operators, more akin to the variation on uniform crossover used in this paper. Single-point crossover seemed to work, but it does mean that a Pokemon is stuck at its index in the population, so perhaps a shuffle operation may prove beneficial. Another area to look into is in fine tuning the parameter probabilities ,which may result in the population converging faster to a better optima. Finally, determining a better way to parse the data for Pokemon available may make the process faster and less erroneous. The way that was used in this paper, is prone to error since I parsed the data and made determinations with my previous Pokemon knowledge to determine the types available to each Pokemon for their moves.

7 Conclusion

In this paper, a genetic algorithm is proposed to create an optimal Pokemon team purely based on typing of Pokemon and those match ups against a predetermined set of opponent Pokemon, here chosen

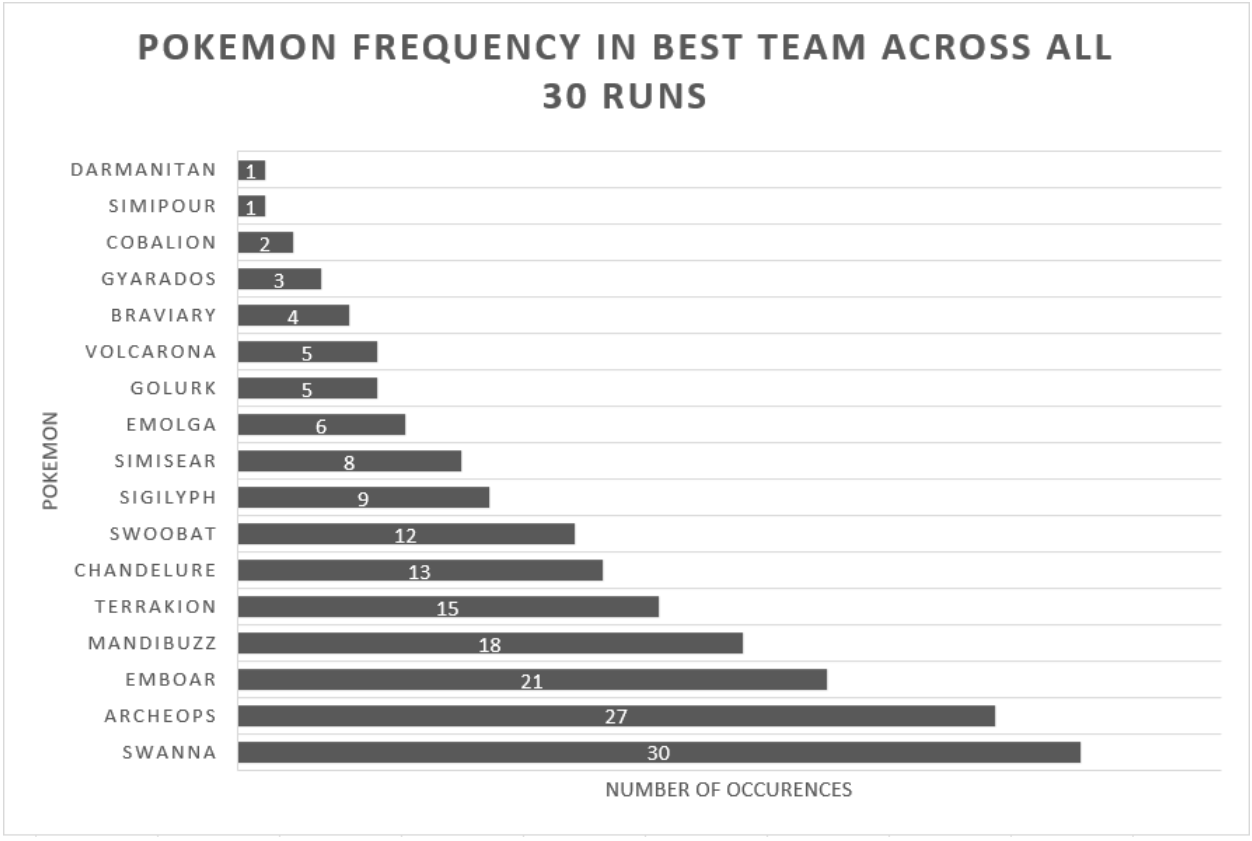

Figure 7: Graph of the number of occurrences of each Pokemon found in the best teams from the 30 runs.

to be the Elite Four and Champion from the games Pokemon Black and White. The Pokemon chosen for the player's team came from a set of 75 Pokemon pulled from Serebii.net along with the types of viable moves that Pokemon can learn. After the set of runs, there were clear standout Pokemon to be chosen to challenge this set of opponents along with the types of the moves those Pokemon should be using. While this algorithm does not produce the global optima consistently, it appears to get quite close every run, improving the fitness of the team from, on average, 205.9 to 383.267 across the 30 runs tested.
This algorithm provides a good starting point to further explore searching for optimal Pokemon teams, and it could also work as the inspiration for future work in other genres of games or other Role-Playing Games. This algorithm can be improved by introduing other objectives and changing the problem to multi-modal and finding a set of best teams.

8 References

References

[1] L. M. Costa, A. C. C. Souza and F. C. M. Souza, "An Approach for Team Composition in League of Legends using Genetic Algorithm," 2019. *18th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), Rio de Janeiro, Brazil, 2019*, pp. 52-61, doi: 10.1109/SBGames.2019.00018.