Bayesian network with Pokémon

Alberto Genovese, Christian Di Buò
Master's Degree in Artificial Intelligence, University of Bologna
{ alberto.genovese5, christian.dibuo }@studio.unibo.it

April 28, 2024

Abstract

This mini-project aims to build and test a Bayesian Network based on the battles of the Pokémon video game. We created two different Bayesian Networks to compare their behaviour and we tested them against online players using the showdown battle simulator. We founded that the bigger network had a worse performance with respect to the smallest one, and both of them can't do equal or better than the classical search algorithm Iterative Deepening.

Introduction

Domain

The Pokémon game is based on battles between two players, in which the objective is to defeat all the opponent player's Pokémon. Every player has a team of six Pokémon, each one with four moves that can either inflict damage or modify the statistics of a Pokémon. During every turn the player can either decide to use one of the four moves or switch the Pokémon with another one in the team. The move can be enhanced or weaken by the type, or the stats, of the Pokémon. Our work take inspiration from the "showdown" project (pmariglia March 2019), from which we have made a fork, in this way we have had the opportunity to use their structure for the bot playing on the online Pokémon battle simulator (Luo December 2011).

Aim

The Objective of this project is to understand the capabilities of the Bayesian Network in predicting the best action in a complex environment, in this case, in a game with few choice, but with a great number of variables. This capabilities will be principally compared to other AI method and we will try different composition of the Bayesian Network, by varying the number of nodes and number of discrete value.

Method

The Dataset, used along the project, has been created by taking around 100 replays of past battles, available as html files in the website showdown (Luo December 2011). It has been written a parser file able to extract the features and automatically create a Dataset as csv file. The Dataset has been subject of the following preprocessing activities:

- continuous features which have been discretized using the KbinsDiscretizer of the sci-kit learn library. (The ranges have been chosen trying to have a good distribution of values)
- aggregation of variables into new variables, such as Enemy Type and Move Type into multiplicator, to indicate the advantage of a move or Pokémon.

To predict the two target variables "choose", for the choice of the move, and "switch" we have built two different Bayesian Networks, using the library pgmpy, the two target can have as value 0 and 1, selected and not selected. We have built two version of the networks one with an higher number of variables and values, one with a lower number to compare their behaviour and performances. In the second version of the Bayesian networks, it has been decided to remove the less important nodes, leaving only the ones that, based on the knowledge of the domain, were the most important. This has been done for both of the networks.

Results

Our exploration reveals that the network, with a big number of variables, gives often a 50% of probability for the choice of the move or the switch, resulting in an inaccurate selection. While the network with the smaller number of variables rarely gives the same problem.

Model

In the BNs built for this study each node represents a unique random variable. These random variables include among the others "Stab", a boolean random variable, which represent the boost given to the move from the types of the Pokémon, "Boost", which is the enhancement, or the weakening, to the stats of Pokémon and "Multiplicator" that represents the relationship between the type of the target move and the types of the opponent's Pokémon. To estimate the Conditional Probability Distributions (CPDs) we have used the Bayesian Estimator. The connections between the variables have been deduced by analyzing the domain: all the variables are pairwise independent, except for "Choose" and "Switch", which completely depends from all the other variables. However the model does not take into account all the possible variables, since we do not have access to all the data of the battles.

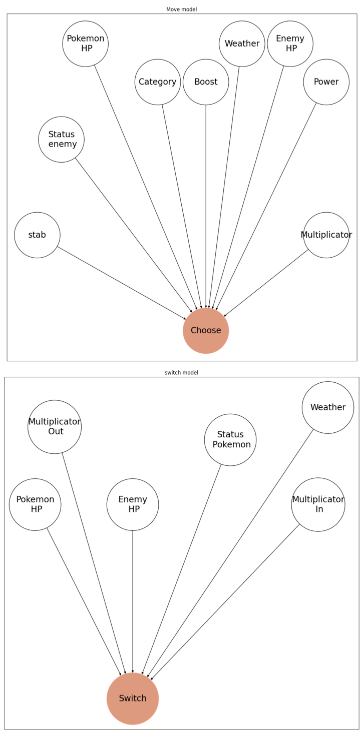

Figure 1: The two Bayesian Network.

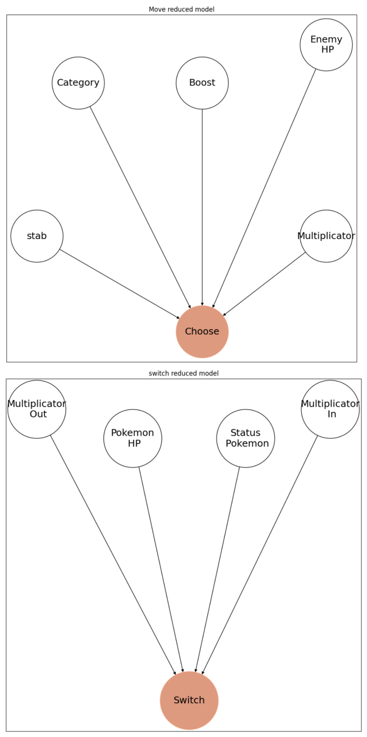

Figure 2: The two reduced Bayesian Networks.

Analysis

Experimental setup

For every move and switch, we run a query on the corresponding BN to predict the ”choose” or ”switch” variable, given the evidence, which is the current state of the game, then the program will select the choice with the highest probability of being the best one. The first comparison method is with a classical AI method for games, Iterative deepening, that was already implemented in the project we forked from (pmariglia March 2019). To assess the results we have decided to create another version of the network, with a smaller number of variables and also some variables with a smaller number of values. We aim to compare these two versions to find out how much the size of the network affects their performances; we let the models play against 100 people in online battles and we compared the result to check which one had the highest number of victory.

Results

In the table has been reported the results of the battles. As we can observe, the best result has been obtained from the Iterative Deepening, followed by the ”Reduced Bayesian Network”, while the worst method has been proved to be the BN with the higher number of variables. This result was predictable since the limited, and not so diversified, amount of data we had could limit the BN favouring the iterative deepening, which do not need a training phase since it is a search strategy algorithm.

	Match	**Win**	**Lost**
Bayesian Network	100	21	79
Reduced Bayesian Network	100	31	69
Iterative Deepening	100	59	41

Table 1: Overview of the results of the battles

Conclusion

After the Analysis of the result, it has been concluded that, in this case, the Bayesian Network with less variable is more accurate than the bigger one. This is probably because the bigger network has a large domain of possible state, this leads every single state to have a lower chance of happening, so the network can’t distinguish well the different situation and tends to give equal probability for every move or possible switch. Because of this the bot tends to choose a random action. Instead the reduced one has less possible state, so the network know well most of them, resulting in more decisive probability for most of the move. However, despite this increase in accuracy, the reduced network still perform worse than the iterative deepening, the cause of this, could probably resides in the structure of the Bayesian networks that can’t describe well the domain.

Links to external resources

The Dataset, the notebook containing the project and the implementation to let the bot play are available on GitHub.

References

[Luo December 2011] Luo, G. December 2011. Pokemon showdown. `https://github.com/smogon/pokemon-showdown`.

[pmariglia March 2019] pmariglia. March 2019. Showdown. `https://github.com/pmariglia/showdown`.