Rutgers University

Artificial Intelligence
01:198:440:01

Assignment: Search and Destroy

Authors:
Azim Khan
Aditya Anikode
Vrinda Jain
Net ids (auk11,aaa387,vj140)

April 13, 2021

RUTGERS
THE STATE UNIVERSITY
OF NEW JERSEY

1 Abstract

In this Assignment we used a combination of knowledge base and probabilistic belief in order to search for a target in a maze with different terrain. Depending on the terrain the search becomes more difficult, however with the use of different belief states, we are able to efficiently direct future actions.

2 Introduction

Our agent is put into a landscape/Maze with various terrains in order to find a target. However, the different terrains make it difficult to search and find the target with the different probabilities and false negatives. As a result the agent might search a spot multiple times. However, with each search the agent gains useful information about the location of the target and is able to narrow down its search. We designed different algorithms for the agent to search the map and built its belief system, in order to find the target efficiently.

3 Project Approach

Before we started working on the code, we familiarized ourselves with the game of search and destroy by playing a couple of games online and also looked up different tips and tricks for the game which could be implemented into our code. Next, we drew up a general rule for our game/environment and what we wanted to include in the game. Based on our environment we explored how we would implement our three different agents into the environment.

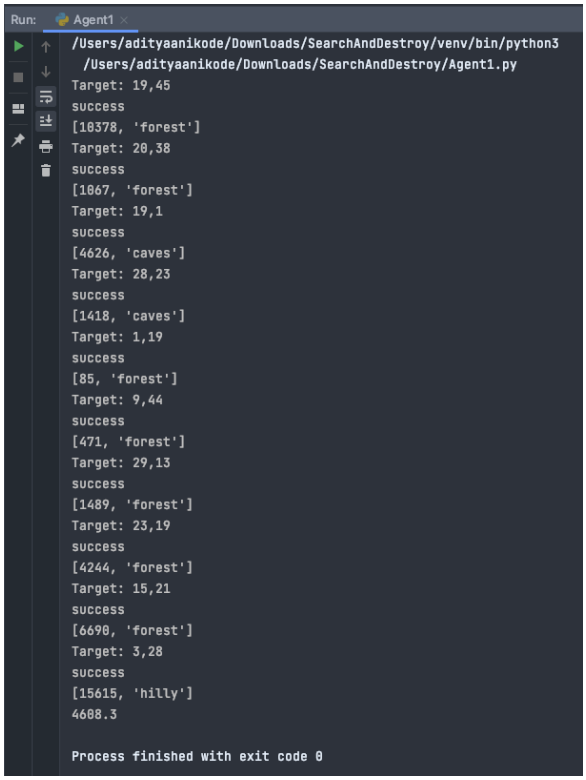
4 Code Explanation

4.1 Environment Class

The environment class is the base of our project where the board is generated given a dimension and number of mines. The init function initializes the board's characteristics: height, width, and terrains. These can be accessed from other functions within the Environment class. We initialize an empty board with no terrains, and then we randomly place the four types of terrains with a 25 percent probability of filling up the board. We also placed a target on a random cell of the board.

4.2 Two Basic Agent Classes

We created two basic agent classes that search for the target and destroy it. Agent one focuses on iteratively travelling to the cell with the highest probability of containing the target, and search that cell. This is done by considering the neighboring cells of the current cell, using belief 1 as well as the the Manhattan distance to see if the cell is closer to the target cell than the prior current cell, to choose the best cell to go to. If it is, this neighbor cell is picked. It would repeat until target is found. The average searches for agent one is 4688.3. However the second basic agent would iteratively travel to the cell with the highest probability of finding the target within that cell, and search that cell. This method uses belief 2, which is problem 2 to determine which next cell to pick. Just like the previous agent, it would repeat until the target is found. The average searches for agent two is 2950.7.



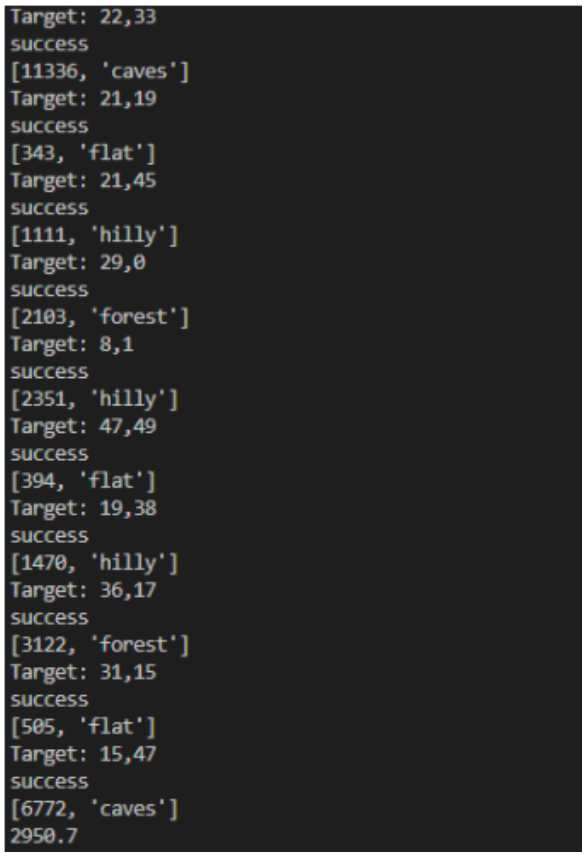


Figure 1: Left: Basic Agent 1 in 50x50 board. The average searches it took to complete was 4688.3 Right:Basic Agent 2 in 50x50 board. The average searches it took to complete was 2950.7

4.3 Detective Perfect Improved K(C)ute Agent Class Helper Upper (Detective P.I.K.A.C.H.U)

```
Target: 43,15
[604, 'hilly']
Target: 30,12
[1345, 'hilly']
Target: 42,23
[1897, 'flat']
Target: 12,12
[152, 'hilly']
Target: 18,12
[1114, 'flat']
Target: 49,4
[8062, 'hilly']
Target: 48,41
[965, 'flat']
Target: 9,8
[1838, 'hilly']
Target: 9,14
[6122, 'caves']
Target: 45,12
[1183, 'flat']
2328.2
```

Figure 2: Improved Agent Agent in 50x50 board. The average searches it took to complete was 2328.2

After making the algorithm for the two basic agents and testing them we have realized that belief state 2 used in basic agent 2 is superior than the other one, so while implementing the improved agent algorithm, we decided to find ways to make it better. We implemented a scoring system into the improved agent to choose the next cell. The Improved agent algorithm decreased the amount of searches compared to the basic agent's algorithm. The score within this method is calculated by dividing Manhattan distance with probability of finding a target in that cell. By using belief 2 for finding the target in a cell, we incorporated Manhattan distance to find the target cell much more efficiently. We have observed that the improved agent is faster in terms of number of searches for terrains with less false negative probability, such as flat terrains and hilly terrains, while taking longer time to search for forest and cave terrains. Through trial and error we have found that the improved agent beats both of our basic agents when we adjust the ratio in terms of our belief of finding the target vs the distance to travel from the current cell, we can drastically improve search efficiency although requiring much more time if we favor our belief over the distance slightly.

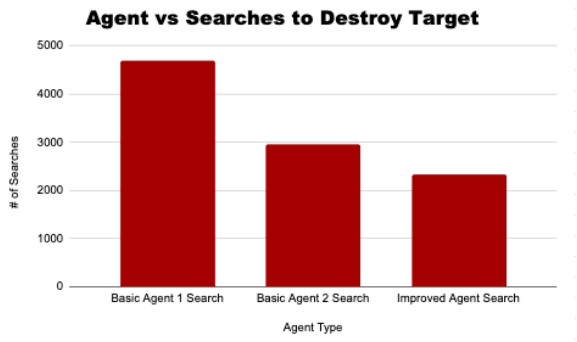


Figure 3: Agent vs of searches taken to destroy target

5 Belief States

5.1 Belief 1 : Probability of Target in Cell

Bayes Theorem is used for calculating conditional probabilities and is used in machine learning. This method is used for calculating the validity of beliefs based on the best available evidence. Depending on the grid, our initial belief stays the same since the target can be anywhere on the map and the agent has no observation or knowledge of any failure of searching the previous cell. So, the agent starts with a random move. Initial belief = 1/ (total number of cells) After the first move the agent now has some information to go off of. Each time a move is made the agent checks if the current cell is the target and if it is present, the agent is done. However if the target is not in the current cell, the agent updates its belief state to be equal to the current cell probability of having the target multiplied by the false negative of the terrain of that cell. Current P(Target in Cell) * P(False negative) If we wanted to find P (Target in Celli|Observationst $\wedge$ Failure in Cellj) at any point First, We would use Bayes Theorem to find P(target in cell | failure in cell) which would be (P(Target in cell) * P(Failure in cell | Target in cell)) / P(Failure in cell) Given this, the agent will recursively update its belief (not the current belief), where the P(target in cell) is the current belief. So in this instance where the agent is looking for the target in a cell given the previous observation (beliefs) and the failure in cell, our current belief simply becomes: Current belief = False negative of terrain * belief of previous cell .

5.2 Belief 2 : Probability of finding target

According to Bayes Theorem P (Target found in Celli| Observationsj) would be:

(P(Target found in Cell) * P(Observations | Target found in Cell)) / P(Observations)

And we know that the probability that a target will be found in a cell is:

P(Target found in cell | Target in cell) * P(Target in cell | Observations)

P(Target found in cell | Target in cell) = (1- False Negative)

So our belief for agent 2 becomes : Belief of previous cell * (1 - False Negative).

Conclusion

Building the game Search and Destroy is a great way to learn how to efficiently model our knowledge/belief about a system probabilistically, and use this belief state to efficiently direct future action. Using Probability is a major area of artificial intelligence and makes decisions based on the data and information that resides in the network. This system helps improve decision making and is useful when when making predictions based on the given information. It also allows us to integrate predictions on a large scale; in our case it would be the 50x50 dimension board. It is capable of predicting new knowledge by using the stored data and making tasks much more efficient and less time consuming. In Search and Destroy, the code utilizes the Manhattan distance to determine which cell is closest to the target cell. By using this data, our agents were able to predict the possible path it can take to reach the target. Basic Agent 1 and 2 required more searches than the Improved agent. In the improved agent, the score is calculated by dividing Manhattan distance with probability of finding a target in that cell. Since Improved Agent made better predictions, it lowered the amount of searches used to find the target, thus helping them win the game. The improved agent is faster in terms of number of searches for terrains with less false negative probability, such as flat terrains and hilly terrains, however it takes longer time to search for forest and cave terrains. Our algorithm was not perfect and there are still several improvements to make.

Contribution statements

In order to divide the work fairly, we decided to make a server on discord where we would meet regularly and work on the project together. We started off by creating pseudo code for each search algorithm and implementing them as we went. Our group as not as proficient as the project needed us to be, so we spent a lot of time learning things on our own in order to write our code.Overall, I believe everyone worked hard and fairly on this project. - Azim Khan

Our group initially struggled with the coding logic for our project. This is because we are all novice programmers and did not have much exposure to Python prior to this project. Therefore, we initially started by making pseudocode for each method and algorithm we were tasked with creating. We spent a lot of time individually learning through youtube tutorials and reading online materials regarding the topics of interest. After individually building our skills, we came together and communicated via discord and messenger video call to bring our ideas to the table. We met regularly, pretty much every day, and diligently worked on the project. Although we were novice coders and struggled initially with the project, we were able to piece it together little by little to form a finished final project.- Aditya Anikode

This project was a challenge. My group met regularly via discord and facebook messenger and collaborated fairly and evenly on this project. We initially struggled with writing the code, so we started by collectively writing pseudo code to better understand the task at hand. Moreover, we spent a lot of time individually learning the concepts and ideas behind each search algorithm, so we could make progress when we met. Overall, I believe everyone put a great deal of effort into this project and the work was divided equally. Although we struggled, a great deal of information was learned. - Vrinda Jain

Honor Code

We have read and abided by the rules of the project, we have not used anyone else's work for the project, and we did all of the work by ourselves.

[1]

References

[1] Wesley Cowan. Assignment 3- probabilistic search (and destroy), 2021.