Intro to AI: Project 3 - Search and Destroy

Ashwin Haridas (ah1058), Ritin Nair (rrn32)

April 2021

1 Introduction

In this assignment, we are tasked with finding a target in a landscape represented by a map of cells. The cells are of various terrain types, and these different types represent how difficult the cells are to search. The target is placed randomly in one of these cells. Given false negative rates, we are asked to compute certain probabilities and then use those probabilities in creating two basic agents and one improved agent. The goal of the agents is to locate the target in as few searches as possible by utilizing the results of the observations collected.

2 Problems

2.1 Problem 1

Given observations up to time t (Observations$_t$), and a failure searching Cell j (Observations$_{t+1}$ = Observations$_t$ $\wedge$ Failure in Cell$_j$), how can Bayes' theorem be used to efficiently update the belief state? i.e., compute:

$$\mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t \wedge \text{Failure in Cell}_j).$$

Solution:

Using Bayes' theorem allows us to compute an unknown conditional probability by using known probabilities. Before we compute the wanted probability, let us list out the known probabilities:

$$\mathbb{P}(\text{Failure in Cell}_i \,|\, \text{Target is in Cell}_i) = \text{Given False Negative Rates} \quad (1)$$

$$\mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t) = \text{Belief}[\text{Cell}_i] \quad (2)$$
where at time $t = 0$, we have $\text{Belief}[\text{Cell}_i] = 1/2500$.

Now, we can first use Bayes' theorem to rewrite the asked conditional probability as:

$$(\mathbb{P}(\text{Observations}_t \wedge \text{Failure in Cell}_j \,|\, \text{Target in Cell}_i) * \mathbb{P}(\text{Target in Cell}_i)) \;/$$
$$\mathbb{P}(\text{Observations}_t \wedge \text{Failure in Cell}_j)$$

We can drop Observations_t (since we always go based off Observations so far) to make things a bit cleaner and in terms that we know. Therefore, our probability formula is:

$$(\mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target in Cell}_i) * \mathbb{P}(\text{Target in Cell}_i)) \,/\, \mathbb{P}(\text{Failure in Cell}_j)$$

Also, we can rewrite $\mathbb{P}(\text{Target in Cell}_i)$ as $\mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t)$. Now, our probability formula is:

$$(\mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target in Cell}_i) * \mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t)) \,/\, \mathbb{P}(\text{Failure in Cell}_j)$$

Before we continue, one thing to note in the numerator is that $\mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target in Cell}_i)$ has two cases to account for, when j = i and when j $\neq$ i.

When j = i, we have:

$$\mathbb{P}(\text{Failure in Cell}_i \,|\, \text{Target in Cell}_i) = \text{Given False Negative Rates}$$

And when j $\neq$ i, we have:

$$\mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target in Cell}_i) = 1$$

At this point, the numerator is in terms of probabilities that we know and the different cases are accounted for. We now have to rewrite the denominator in probabilities that we know. We can do this by marginalizing $\mathbb{P}(\text{Failure in Cell}_j)$ over the different disjoint cases in which a failure can occur. After marginalizing, we have:

$$\mathbb{P}(\text{Failure in Cell}_j) = \mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target not in Cell}_j) * \mathbb{P}(\text{Target not in Cell}_j) + \mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target in Cell}_j) * \mathbb{P}(\text{Target in Cell}_j)$$

We know that $\mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target not in Cell}_j) = 1$, therefore:

$$\mathbb{P}(\text{Failure in Cell}_j) = \mathbb{P}(\text{Target not in Cell}_j) + \mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target in Cell}_j) * \mathbb{P}(\text{Target in Cell}_j)$$

After doing some additional cleanup to write things in probabilities that we have...

$$\mathbb{P}(\text{Failure in Cell}_j) = (1 - \mathbb{P}(\text{Target in Cell}_j \,|\, \text{Observations}_t)) + \mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target in Cell}_j) * \mathbb{P}(\text{Target in Cell}_j \,|\, \text{Observations}_t)$$

Now we have everything in terms of probabilities that we know. Our final probability is then:

When j = i:

$$\mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t \wedge \text{Failure in Cell}_i) = (\mathbb{P}(\text{Failure in Cell}_i \,|\, \text{Target in Cell}_i) * \mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t)) \,/\, ((1 - \mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t)) + \mathbb{P}(\text{Failure in Cell}_i \,|\, \text{Target in Cell}_i) * \mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t))$$

When j $\neq$ i:

$$\mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t \wedge \text{Failure in Cell}_j) =$$
$$(\mathbb{P}(\text{Target in Cell}_i \,|\, \text{Observations}_t)) \;/$$
$$((1 - \mathbb{P}(\text{Target in Cell}_j \,|\, \text{Observations}_t)) +$$
$$\mathbb{P}(\text{Failure in Cell}_j \,|\, \text{Target in Cell}_j) * \mathbb{P}(\text{Target in Cell}_j \,|\, \text{Observations}_t))$$

2.2 Problem 2

Given the observations up to time *t*, the belief state captures the **current probability the target is in a given cell**. What is the probability that the target will be **found** in Cell_i if it is searched:

$$\mathbb{P}(\text{Target found in Cell}_i \mid \text{Observations}_t)?$$

Solution:

We can start my marginalizing the given probability over the different disjoint cases in which it can occur. This allows us to use probabilities we already know to calculate the unknown probability in this problem. We start by marginalizing by the cases in which the target is found in the cell. We can consider the two disjoint cases where this could potentially happen as being if the target is found in the cell and the target is in the cell, and if the target is found in the cell and the target is not in the cell. This is modeled by the following equation:

$$\mathbb{P}(\text{Target found in Cell}_i \wedge \text{Target in Cell}_i \,|\text{Observations}_t) +$$
$$\mathbb{P}(\text{Target found in Cell}_i \wedge \text{Target not in Cell}_i \,|\text{Observations}_t)$$

We can then use conditional factoring on the expression to pull out the probability of target being in the cell and the probability of the target not being on the cell to further simplify these two terms. This is shown by the equation:

$\mathbb{P}(\text{Target in Cell}_i \mid \text{Observations}_t) * \mathbb{P}(\text{Target found in Cell}_i \mid \text{Observations}_t, \text{Target in Cell}_i) +$
$\mathbb{P}(\text{Target not in Cell}_i \mid \text{Observations}_t) * \mathbb{P}(\text{Target found in Cell}_i \mid \text{Observations}_t, \text{Target not in Cell}_i)$

The probability of a successful search when at a Cell_i given that the target is actually in Cell_i is independent of the Observations_t taken so far since the chance of the search being successful in this case then relies only on a false negative not occurring. Similarly, the probability of the target being found in Cell_i given that the target is not in Cell_i is also independent of the Observations_t taken so far. This is because the false positive rate is given in the description of the assignment to be 0. In other words:

$$\mathbb{P}(\text{Target found in Cell}_i \mid \text{Target not in Cell}_i) = 0$$

This holds true regardless of the Observations_t taken so far, so:

$$\mathbb{P}(\text{Target found in Cell}_i \mid \text{Observations}_t, \text{Target not in Cell}_i) = 0$$

We can use the concept of the Observations_t being independent of a successful search if it is given that the target is in Cell_i and the fact that the false positive rate is always 0 to further simplify the equation:

$$\mathbb{P}(\text{Target in Cell}_i \mid \text{Observations}_t) * \mathbb{P}(\text{Target found in Cell}_i \mid \text{Target in Cell}_i) + \mathbb{P}(\text{Target not in Cell}_i \mid \text{Observations}_t) * 0 = \mathbb{P}(\text{Target in Cell}_i \mid \text{Observations}_t) * \mathbb{P}(\text{Target found in Cell}_i \mid \text{Target in Cell}_i)$$

We also know that:

$$\mathbb{P}(\text{Target found in Cell}_i \mid \text{Target in Cell}_i) = 1 - \mathbb{P}(\text{Target not found in Cell}_i \mid \text{Target in Cell}_i)$$

Therefore, we can reasonably conclude that:

$$\mathbb{P}(\text{Target found in Cell}_i \mid \text{Observations}_t) = \mathbb{P}(\text{Target in Cell}_i \mid \text{Observations}_t) * (1 - \mathbb{P}(\text{Target not found in Cell}_i \mid \text{Target in Cell}_i))$$

2.3 Problem 3

Compare agents 1 and 2. Which agent is better, on average?

Solution:

When attempting to locate the target, Agent 1 iteratively travels to the cell with the highest probability of the cell containing the target. Agent 2 iteratively travels to the cell with the highest probability of finding the target within that cell. These probabilities are a bit different, as explained in problems 1 and 2. After running 100 trials per agent, we found that the average score for Agent 1 was approximately 35,000 and the average score for Agent 2 was approximately 30,000. Based off these scores, it seems, that Agent 2 is better than Agent 1. However, it turns out that the score is dependent on what terrain the target is placed in. In general, after doing some additional tests, when the target is placed in a Flat / Hilly terrain cell, Agent 2 is always scoring better (by a good amount) than Agent 1. But, when the target is placed in a Forested / Cave terrain cell, there are a few times when Agent 1 is actually scoring better (by a little) than Agent 2. This was a bit surprising to us, but after looking more closely at the probabilities we are dealing with, we may have a justification as to why this is the case, as well as to why on average Agent 2 is better than Agent 1.

The probability that Agent 2 uses to find the cell to search next is the current belief that the target is in the cell multiplied by the belief that the target is found in the cell given if the target is in the cell. This second belief is based entirely on the false negative rates given to us from the project description. What this means is that our Agent 2 will prioritize visiting cells that have a low false negative rate, which in our case are cells that are in Flat / Hilly terrains. Therefore, if the target is in a Flat / Hilly terrain, Agent 2 will probably find the target relatively faster than Agent 1. This also means that Agent 2 will *not* prioritize visiting cells that have a high false negative rate, which in our case are cells that are in Forested / Cave terrains. This means that if the target is in this type of terrain, Agent 2 will likely take much more time to find the target than if the target was placed in the other terrains. In fact, in this scenario, it seems that Agent 2 has around the same score (sometimes even a bit worse) than Agent 1 when the target is placed in the Forested / Cave terrains because of the low priority these terrains are given. Agent 1 does not really prioritize based off the FNR, which may explain why in the Forested / Cave terrains, Agent

1's score is close and sometimes better than Agent 2's score.

Overall, the improvement that Agent 2 has compared to Agent 1 when the target is placed in a Flat / Hilly terrains is significantly larger than the slight advantage that Agent 1 has over Agent 2 when the target is placed in the Forested / Cave terrains. After a large number of trials, this results in the net advantage that Agent 2 has over than Agent 1 to be greater, which explains our results of the better average score Agent 2 has.

2.4 Problem 4

Design and implement an improved agent and show that it beats both basic agents. Describe your algorithm, and why it is more effective than the other two. Given world enough, and time, how would you make your agent even better?

Solution:

After running 100 trials with our improved agent, we were able to get a score of approximately 9,000, which is significantly better than Basic Agent 1 and Basic Agent 2. There are two main improvements that our improved agent utilises to beat both of the basic agents on average. The first improvement is considering the distance it takes to travel to a new cell in addition to the probability of finding the target at a new cell given the observations at time t. The second improvement is not simply searching each cell once, but searching it multiple times to account for cells where the terrain results in higher false negative rates.

Improvement 1:

Based on the answers to problem 3, we can safely conclude that agent 2 performed better than agent 1. Agent 2 uses the probability of the target being found in a cell given the observations found as a decision making mechanism to select the next cell to be searched. This approach, however, does not account for the cost of moving from one cell to another before a search can be performed. If the cell with the highest probability of the target being found given the current observations at time t is across the board from the current cell, then the score significantly increases based on the distance that it takes to travel from the current cell to the selected one. Due to agent 2 not accounting for the distance it takes to travel to another cell when deciding which cell to search next, the score on agent 2 ends up being relatively high when compared to our improved agent.

The improved agent accounts for the distance as well as the probability when deciding what the next cell to search will be. In order to find the target in the least number of searches possible we want to select the cell with the highest probability of the target being found given the observations at time t, but we also want to prioritize selecting a cell with a low distance to travel to from the current cell. We can model this idea with a new matrix in which we have each cell contain a ratio between the probability of the target being found at the cell given the observations at time t and the distance it would take to travel from the current cell to the next cell. In other words, each cell in the this new ratio matrix is assigned the value:

$$\text{Ratio} = \mathbb{P}(\text{Target found in Cell}_i \mid \text{Observations}_t) \; / \; \text{Distance from Cell}_{curr} \text{ to Cell}_i$$

In this formula, the current cell the agent is on is Cell_{curr} and each cell on the map is Cell_i. The only exception to this ratio assignment is if $\text{Cell}_i = \text{Cell}_{curr}$, in which case the ratio is assigned a value of 0. This is to make the algorithm more likely to not stay on the same cell since the second improvement already accounts for searching one cell multiple times before moving to a new one.

The next cell to search is the one that has the highest value for this ratio, since that means it has a relatively high probability and a low distance cost when compared to other cells on the board. While we do want to pick cells with high probabilities of the target being found given the observations so far, we do not want to do so at the cost of a high distance to travel from the current cell to the new cell. This approach seeks to remedy this issue by factoring in the cost of distance in the process of selecting which cell to search next. Therefore, the score of the improved agent ends up being lower than the basic agents because it takes distance into account more than the basic agents do.

Improvement 2:

Agent 1 and Agent 2 both search a cell once before updating their belief systems and choosing the next cell to search. It is often the case that the next cell to search is not the same as the current one, thus resulting in false negatives making the agent miss the target. We can make an improvement to these agents by accounting for false negative rates more than the basic agents do. The main idea is that in terrains with higher false negative rates the improved agent benefits from conducting more searches on the current cell to lessen the impact of a false negative. For flat terrain only one search is performed, for hilly terrain two searches are performed, for forested terrain four searches are performed, and for cave terrain ten searches are performed. The reason we chose these values is because we want the expected number of searches where a false negative did not occur out of all the searches on the current cell in one iteration of the algorithm to be close to one. This means there was around at least one search that did not fail due to a false negative. This concept can be modeled by the following formula:

$$(n) * (1 - fnr) = E(s)$$

In this formula n is the total number of searches performed on the cell, fnr is the false negative rate, and E(s) is the expected number of searches that did not fail due to a false negative. We tried to pick an n value for each type of terrain such that E(s) was close to 1. Taking this approach increases the likelihood of not simply moving away from a cell after a search that failed due to a false negative. In cases where this happens in the basic agents, the score significantly increases due to the distances of traveling to new cells and eventually back to the one with the target. This improvement seeks to fix this problem and helps the improved agent score better than the basic agents.

Algorithm:

The improved agent first initializes the belief state in the same way basic agent 1 does (denoted as the belief_matrix) and the matrix with the probability of the target being found in each cell given the observations (denoted as the found_matrix). A random cell is selected first to search. After the cell is searched, on success the score is returned and on failure the algorithm continues. The belief_matrix and found_matrix are then both updated. The found_matrix is used to construct a matrix containing the ratio of the probabilities in the found_matrix of a cell to the distance to travel to that cell from the current cell. These ratios are used to make a new matrix (denoted as the ratio_matrix) which is used in the decision making process for selecting the next cell to search.The next cell to be searched is the one with the highest corresponding ratio value from the ratio_matrix. Then, the following process is repeated until the target is found. Depending on the current cell's terrain, a certain number of searches are performed. For flat terrain only one search is performed, for hilly terrain two searches are performed, for forested terrain four searches are performed, and for cave terrain ten searches are performed. When multiple searches are performed, the belief_matrix and found_matrix are also updated to account for an additional failure at the current cell being observed. If all these searches fail, the ratio_matrix is recalculated based on the updated found_matrix. It is then used to decide the next cell to travel to. When the target is found the total searches + distance traveled are returned as the score for the improved agent.

Improvements With Infinite Time:

Given infinite time we could further improve the calculations in our ratio matrix. Currently, we take the ratio between the probability of the target being found given observations at time t and the distance from the current cell to the next potential cell. The reason we use this distance value is because we want to minimize the distance cost to get a better score. We can improve on this idea by not only simply trying to minimize the immediate cost, but also trying to minimize based on an estimation of future distance costs. This is similar to how Markov Decision Processes, or MDPs work. By accounting for future costs, we can more accurately pick a cell to travel to that is also likely to minimize the costs in the future as well. The specific process to do this would be to first find the immediate cost of moving from Cell_{curr} to Cell_{t1}. Then, once at Cell_{t1} we need to consider what happens in the event of a failed search. We would need to update a copy of the belief_matrix and the found_matrix to account for the hypothetical failed search at Cell_{t1}. Then, we would create a new ratio matrix based on the belief state of this hypothetical situation where a failure occurred at Cell_{t1} to model what cell would be selected to travel to after the failure at Cell_{t1}. This new ratio matrix would only use one distance value in the denominator for ratio calculations and would not account for potential future distance costs in order to avoid infinite looping from the ratio matrix calculations. This is to say that it would use the formula:

$$\text{Ratio} = \mathbb{P}(\text{Target found in Cell}_{t2} \mid \text{Observations}_{t+1}) \;/\; \text{Distance from Cell}_{t1} \text{ to Cell}_{t2}$$

In this formula Cell_{t2} represents each cell on the board that is considered as the next potential cell to move to from Cell_{t1}. Note that we don't account for future costs in the ratio calculations for the next cell to search after Cell_{t1} to avoid infinitely looking into the future with no certain end. This ratio matrix would be used to determine the next cell to go to after failing at Cell_{t1}, which we can denote as Cell_{t2}. The distance from Cell_{t1} to the next cell, Cell_{t2}, multiplied by the probability

of this failure occurring would also be added to the denominator of the ratio calculation when determining the next cell to search after failing at Cell_{curr}. This can be shown by doing:

$$\begin{gathered}\text{Ratio} = \mathbb{P}(\text{Target found in Cell}_{t1} \mid \text{Observations}_{t}) \ / \\ (\text{Distance from Cell}_{curr} \text{ to Cell}_{t1} + \\ (\mathbb{P}(\text{Target not found in Cell}_{t1} \mid \text{Observation}_{t+1})) * \\ (\text{Distance from Cell}_{t1} \text{ to Cell}_{t2}))\end{gathered}$$

In this formula Observation_{t+1} uses a copy of the current belief_matrix and found_matrix to generate a belief state which accounts for the hypothetical failure at Cell_{t1} as well. Using this formula helps account for potential future costs incurred from traveling to new cells upon failure, which can result in a better choice for a next cell which minimizes this cost into the future as well. This approach takes significantly more time, however, as multiple new matrices need to be generated at each step when predicting the future of the agent. Also, this approach accounts for additional costs one step into the future, but given infinite time we could account for n steps into the future.

A further improvement that allows predicting n steps into the future would be to not stop the ratio matrix calculations at accounting for a failure at Cell_{t1} just one step ahead, but rather at Cell_{tn} which would account for n failures into the future. This means every intermediate ratio matrix calculation would need to account for the future, until the one which has picking $\text{Cell}_{t(n+1)}$ to travel to from $\text{Cell}_{tn)}$ which would act as a terminating case to get a use-able calculation. Note that the individual ratio matrices are not predicting n steps ahead of their individual respective time steps, but rather they all predict until step (t+n) which is n steps ahead of the current time. The formula for this would be:

$$\begin{gathered}\text{Ratio} = \mathbb{P}(\text{Target found in Cell}_{t1} \mid \text{Observations}_{t}) \ / \\ (\text{Distance from Cell}_{curr} \text{ to Cell}_{t1} + \\ \mathbb{P}(\text{Target not found in Cell}_{t1} \mid \text{Observation}_{t+1}) * \\ \text{Distance from Cell}_{t1} \text{ to Cell}_{t2} + \ldots + \mathbb{P}(\text{Target not found in Cell}_{tn} \mid \text{Observation}_{t+n}) * \\ \text{Distance from Cell}_{tn} \text{ to Cell}_{t(n+1)} \ )\end{gathered}$$

By using this ratio as the value in the ratio matrix to determine what the next cell to search after the current one would be, the improved agent is able to account not only for immediate costs, but rather for all distance costs incurred from n time steps into the future. Doing so allows the improved agent to select a sequence of cells to search that is likely to improve its total score over the course of the whole search process.

3 Contributions

In this project both members attempted to divide up the responsibilities as evenly as possible. Ashwin implemented a significant portion of setting up the board, including generating cells with terrain types and placing the target in the board. He also implemented basic agent 1, in addition to the associated functions necessary to complete the agent. This included finding the next cell to search when given a matrix with values to prioritize by, and breaking ties by distance. Ritin then

used the framework provided to implement basic agent 2 which involved making minor additions to the implementation of basic agent 1 to account for the different requirements of the second agent. Ritin also worked on implementing the improved agent which involved adding on the framework provided by the implementation of basic agent 2. Both members worked on problems 1 and 2 in the report to derive the formulas to be used later on in the project. Ashwin primarily focused on problem 1 and Ritin mainly focused on problem 2. For the final write-up Ashwin did problems 1 and 3 and Ritin did problems 2 and 4.

4 Honor Code

This assignment was done on our own. None of the code or the report was copied or taken from online sources or any other student's work.