Lab Assignment 2: Where's Croc

Lab Group 31

Wikner, Anton

Lee, Jing He

Kindvall, Kalle

Wiengarten, Lisanne

1. Hidden Markov Models (HMM)

Theoretical overview

A Hidden Markov Model (HMM) is a temporal model used to represent probability distributions over sequences of observations. Given a sequence of observations and without knowledge of the sequence of states the model went through, we would be able to derive the probabilities of the sequence of states through analysis of the observed data. This sequence of observations consists of observation taken at multiple time slices, such that an observation at time *t* only depends on the states at time *t*.

Some key terms involved in HMM are:

Emission Probability Matrix:

Each value in the matrix represents the probability of getting a particular observation from a state within current time slice.

Transition Probability Matrix:

Each value in the matrix represents the probability of getting from a state in *t-1* to a state in *t.*

Refer to the diagram below for a simplified version where each state is represented by only one value. For example S_1 represents the state at time $t = 1$. In this diagram, the probabilities would be given as follows:

$$\text{Emission probability} = P(O_t \mid S_t),$$

$$\text{Transition probability} = P(S_t \mid S_{t-1})$$

$$\begin{tikzcd}
S_1 \arrow[r] \arrow[d] & S_2 \arrow[r] \arrow[d] & S_3 \arrow[r] \arrow[d] & \cdots \arrow[r] & S_n \arrow[d] \\
O_1 & O_2 & O_3 & & O_n
\end{tikzcd}$$

Current state estimation algorithm

Given a sequence of observations and an initial state, the probabilities for the state of the system at a specific time can be obtained. Below, one can see the same simplified diagram where each state is represented by only one value but with the initial state included, S_0.

To find the probabilities for the state at a specific time, the forward-backward algorithm has to be employed. The forward part of the algorithm involves deriving the probabilities for the current state from the state at earlier times while the backward part of the algorithm involves deriving the probabilities for the current state from the state at later times. For example, finding the probability of S_2 involves moving forward from S_1 and moving backwards from S_3.

$$\begin{tikzcd}
S_0 \arrow[r] & S_1 \arrow[r] \arrow[d] & S_2 \arrow[r] \arrow[d] & S_3 \arrow[r] \arrow[d] & \cdots \arrow[r] & S_n \arrow[d] \\
& O_1 & O_2 & O_3 & & O_n
\end{tikzcd}$$

2. Hidden Markov Models (HMM) in our script

The HMM we used in our script involves only the forward algorithm for state estimation as we are only interested in the latest location of Croc, so there is no observations available from later time slices for backward propagation. The observations, probability matrices and initial state are as such:

Observations:

The observation for each time slice is taken from the water condition reading recorded by the sensor on Croc. Thus, at each time slice we obtain the three values for calcium, salinity and alkalinity at Croc's location.

From these values, we want to calculate which waterhole is Croc's most probable current location. We do so by comparing the values around Croc to the values we get for all waterholes from the sensors installed there.

Transition probability matrix:

Since the Croc moves or stays still randomly, the transition probability for each waterhole is determined by the number of neighbouring waterholes they are connected to. A waterhole with 4 neighbouring waterholes would have a value of $\frac{1}{5} = 0.2$ for transition to these 4 neighbouring waterholes and to itself. The rest of the transition probability to other waterholes would be 0.

For example, W_1 has 4 neighbouring waterholes W_2, W_3, W_4, W_5. Then the values in the transition probability matrix for $w_{1,1}$, $w_{1,2}$, $w_{1,3}$, $w_{1,4}$ and $w_{1,5}$ would be 0.2 and the rest (e.g. $w_{1,6}$) would be 0.

Plus, the state of the two backpackers can be considered as well. If we observe that one of the backpackers died, we know that Croc is currently at this backpacker's location. Therefore, we can set all transition probabilities for this time slice to 0, except for the neighbours of the waterhole where the dead backpacker is at, because Croc can only move to one of those neighbours (or stay at the same waterhole).

Emission probability matrix:

Using the `dnorm()` function and the distributions for calcium, salinity and alkalinity of each waterhole, we calculated the emission probability by multiplying the return value from `dnorm()` for the 3 water conditions of each waterhole. For clarity, this means:

$$emission\ probability = dnorm()\ for\ calcium \times dnorm()\ for\ salinity \times dnorm()\ for\ alkalinity$$

Multiplication is used as this preserves the order of magnitude for each return value, as opposed to using addition which could unbalance the weight of each water condition.

Initial state:

Since Croc could be at any waterhole except where the hikers are, 38 waterholes are assigned equal probability as their initial state which is taken as $\frac{1}{38} = 0.0263$. The waterholes where the two hikers start are assigned initial state probabilities of 0.

3. Our additional strategies

After the Hidden Markov Models have been used to give us a good estimate of where the crocodile was we move towards that node. This is not completely trivial since the nodes are not connected only in a line with higher and lower values on either side. So to find the shortest route from one node, our current node, to the goal node, the node where we think that the crocodile was, we had to create a separate function.
It is simple in the sense that it does not take into account potential places along the way that could be more flexible and therefore potentially more advantageous. This is one way that the code could have been improved and it might have made the entire algorithm more effective. When given the choice between two nodes that goes towards the goal in the same amount of steps, then most connected node could be better.
Another such consideration could be to pick a node that is also closer to the second most likely node where the crocodile could be. Neither of these tactics are implemented. Instead, the route-finding algorithm simply expands outwards from the starting node to all new nodes that are connected.
This is then continued until the goal is found, and as soon as the goal is found, it must be the shortest route and the goal can be found from several different paths in the same expansion-round. In such a scenario the route which is found last simply overwrites the others and then the route is found by looking back through the steps taken to get there.

We did try some other ways to improve the performance. By also looking at the second most likely position for the crocodile to be, we get some more opportunities to make smart decisions. When the two nodes were very far apart, we tried to walk towards the middle of them instead of getting locked in on one.
This did impact the performance negatively, which could be because as discussed above, the nodes are not connected only based on high and low numbers and thus to take advantage of this it would need to be included within the pathfinding algorithm.
Another reason that this might not have been beneficial is that the relative confidence of the first and second guess was not taken into account. Perhaps this tactic is best to use only in the first few steps since after a few steps the crocodile is most likely close to being caught with an average of about 7 rounds.

One thing that did make a difference was when to look for the crocodile. After trying different combinations of walking and looking we settled on generally first walking and then looking. This gave a better performance, one factor probably being that the grid is relatively small and many games ending in just a few rounds where the crocodile might even be caught before the algorithm has had time to truly zero in on it’s position and the crocodile happened to have walked towards the hunter and was closer than expected.

There was one scenario when it was more beneficial to look first, and that was when the crocodile was expected to have been one node away in the previous turn. If the crocodile walks towards the hunter, then it is caught, and if it walks away, then the hunter is still moving towards it.