HW 07- Hidden Markov Models

08/06/2020

ISHAN JOSHI

USC ID 3451334123

Introduction

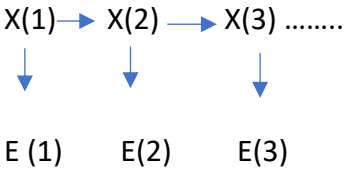
Hidden Markov Models

A markov model is a chain of events, where a variable is repeated at different time steps. The For eg:

X (1) → X (2) → X (3) → ….

The above graphical model can be denoted in terms of two Conditional Probability Tables (CPTs). One denoting the relationship between X(t) and X(t-1) and the other denoting the initial probabilities X (1). These models have the Stationary Property, which mains that at any point of time in the chain, the value of P(X(t)) depends only on the value of P(X(t-1)), even if t tends to infinity. A popular application of markov models is the google page rank algorithm.

Hidden markov models consist of two nodes on every timestep. There are hidden variables X(t) and evidence variables E(t) which is visible to the user as an observation sequence.



This graphical model contains three CPTs:

1. Between X(t) and X(t-1)
2. X(t) and E(t)
3. X(t)

An application of HMMs is Robot Localization, where the hidden variables could correspond to the true position of the robot at a point in time while the evidence variables are the various signals that we receive from the sensors attached to the robot.

The forward propagation of the algorithm involves the creation of the above CPTs, based on which we can translate the entire graphical model into simply a collection of tables. All future computation then involves only the tables created.

Viterbi Algorithm

Viterbi algorithm is a dynamic programming algorithm that is used to decode the hidden markov model and find the most likely values for the hidden variable for a given observation sequence.

Member Functions

1. **Multiply_xt_xtprev():** creates the table of interactions between the state variable X at timestep t and t-1. The function returns the resultant 10x10 matrix.
2. **Multiply_Xt_and_Et():** creates the table of interactions between the state variable X and its evidence variable E at timestep t. The function returns the resultant 12x10 matrix.
3. **Main_function():** performs the main execution of the algorithm. It calls the multiply_xt_xtprev function to get the interaction table between X(t) and X(t-1). It then multiplies each row of the above table with P(X(t-1)). It then calls the multiply_Xt_and Et function to get the interaction table between X(t) and E(t), and the row corresponding to the observation sequence is multiplied to the previous result. The projection then gives us the final index or most likely sequence.

Code Challenges

1. I created the variable interaction tables for timestep 1 before creating the above function structure, to make the subsequent calculations more convenient.
2. Due to the process adapted above, an additional conditional check was needed in the loop in the main function.

Code Output

The given observation sequence is:

```
[8, 6, 4, 6, 5, 4, 5, 5, 7, 9]
```

The most likely sequence of values for the variable X are as follows:

[7. 6. 5. 6. 5. 4. 5. 4. 7. 8.]