9418 Assignment2

Qiwen Zheng[1] and Hang Zhu[2]

[1]z5240149
[2]z5233612

1 Basic Idea

After reviewing papers (attached on Reference), we come up with the algorithm that uses Markov chain to achieve dynamic crowd detection as time flow.

First thing we did is to search the data.csv and find out possible next step of each space (space including r1-35,c1-4,o1,outside, 41 in total) and store them into a dictionary, we called it outcomeSpace. Then we calculate the transition matrix since we need to model the possible next step of each space to make statistics on people movement at each time slot. The transaction matrix is a 41 by 41 matrix in which each value represents the possibility of row numbered space transfer to column numbered space. How to calculate the matrix is introduced in Assumption part.

We record the number of people at each room using a Numpy defined array, called state. Because of the fast matrix multiplication of Numpy package. We can do multiplication of state and transaction matrix. Therefore we get the next time possible people number in each space and do inference according to it.

Since the transaction matrix is a probability matrix, the people number can be any real number greater than or equal to 0. We set a threshold 0.2, every 15 seconds if the number is beyond the threshold, we decide to turn on the light since the cost of not turning on the light but someone in there is 4 times the cost of turning on the light but no one there. We follow the strategy of turning on as much light as possible unless there is quite low possibility there is person in a room.

Also we use the strategy of "inhomogeneous Markov chain", which is a time based Markov chain that different time period has different Markov transaction matrix. We create three different Markov chains, respectively from 8 to 8:05 (go to work time), 8:05 to 17:30 (work time), 17:30 to 18:00 (go off work time).

The sensors (4 reliable sensors, 4 unreliable sensors, 4 door sensors and 2 robots) in our model are used to adjust the number of people within each space and can also avoid the Markov chain convergence problem. Every 15 seconds

after the matrix multiplication, we get the new state of each room, and we use the sensor information from function parameters and do some adjustment. For example, the reliability of two robots is 100 percent so what data we get from the robots can be directly used to set the number of people in the state variable it observe.

The reliability of other sensors is not 100 percent and they can not tell the number of people in the space, either. Our strategy is to calculate the reliability of other sensors (4 reliable sensors, 4 unreliable sensors) based on the TP and FN rate. If the reliable or unreliable sensors get motion but the number of people in the space the sensors located is below threshold, we lift up the value of the space to the TP rate of the sensor. Similarly, if the sensors get no motion but the number of people in the space the sensors located is over threshold, we lower down the value to the FN rate of the sensor. So at that moment, the light would turn on or off according to the sensor motion state, but the value reset is going to pose an influence the afterwards state. Specifically, we found that the TP rate of unreliable sensor 4 is quite low, around 25 percent, so we ignore the sensor.

For the four door sensors, we also calculate the theirs' "reliability". But this time we only record the TP rate, which relates to the both sides of the door when door sensors get people passed through. If there is somebody who goes through the door sensor, and the space on either side of the door has under threshold value, we lift value up to the TP rate of the door.

2 Assumption

Based on our model,we make the following assumption.

1. The reliable and unreliable sensors are only related to their' located space, which means the information we get from the sensors only indicates if there is somebody in the space where the sensor locates. Whereas the door sensors are only relevant to their' two sides space.

2. If the number of people of one space does not change compared to the last time step, we assume there is no people flow happened on the space.

3. There is no limitation of how many steps one person can move within 15 seconds, we only concentrate on the number of people variation and we made outcome space based on the possible next step of each space.For example, if the number of people at r1 at one time is 3, next time the number deducted to 2, all other rooms' people numbers don't change except o1's number plus one. Theoretically there is no way one person can move to o1 from r1 within 15 seconds,the true people flow may be one person goes to c1 from r1, one person goes to c2 from c1, then one person goes to o1 from c2. But we only make assumption that there is one person moves to o1 from r1,and we add o1 to the outcomeSpace of r1.

4. Within each space variable of outcomeSpace, the order of the possible next step space matters.T he smaller the index of the next step space, the more likely the people will move to it from the current space (usually the smaller index

space is more closed to the current space). The computation of transaction rate is highly based on the order.For instance, when we compute the transaction rate of r1, we dive into the data set, if at one time the number of people in r3 is 3, next time the number reduced to 2, we firstly doubt the one person moves to r1, if r1's number increases by 1 or more, then we record the transfer to the r1, if r1's number stays still, then we look at r7 and so on so forth.

5 .The essential part of the project is calculating the Markov transaction matrix.we use the formula shown as followed:

$$Pij = \frac{Tij}{Si}(i! = j) \tag{1}$$

where Tij is the number of runs to j from i when i decreases during the specified time period,Si is the sum of all the people of space i during the specified time period.

$$Pii = 1 - \frac{Di}{Si} \tag{2}$$

where Di is the sum of all the number of deduction from i during the specified time period,Si is the sum of all the people of space i during the specified time period.

using above two formulas, we are able to build up transaction matrix.

3 Programming Design Process

• The get_pa.py file generates csv file that records the three time period Markov transaction matrix probability.
• The reliability.py file prints out the reliability (TP, FN) of sensors and robots.
• The solution.py file gets the three time period transaction matrix and initializes "state" to be a list with all 20 people be outside. Then within the get_action function, we multiple the last time state with the time-based transaction matrix, afterwards, we use the evidence (the information of sensors and robots) to revise the value within the state. Eventually based on the values of the state, we are able to turn on the light if the corresponding state is over the threshold.

4 Efficiency issues

• The get_pa.py: O(M^2N), where M is the number of all spaces (41 for this case), N is the number of whole training set(2400 for this case).
• The reliability.py: O(SN), where S is the number of all sensors (14 for this case), N is the number of whole training set(2400 for this case).
• The solution,py: After reading in the transaction matrix, our get_action function can run very fast, this is because the fast calculation of matrix multiplication by Numpy @ operator, which is implanted in C and processes the

multiplication of matrix parallelly. Also the sensor adjustment part is all "if" statement, so it is O(1) time complexity altogether of one run.

Based on the Leaderboard run test, our running time is so short that even closed to the baseline case, which is 35 seconds of 10 runs.

Assignment 2 Leaderboard

Posted by Jeremy Gillen 4 days ago.

zID	Cost	Time(s)
z5240149	59003	36

5 Overfitting Problem

We tried five time period Markov chain(8:00 - 8:05, 8:05 - 11:00, 11:00 - 13:00, 13:00 - 17:30, 17:30 - 18:00) and implement it on the newest and last chance of Leaderboard. According to the test result of 10 day runs which shown below, it is getting worse and gets even higher average cost, this is because the overfitting problem that we cut so many time period. So we decide to switch back to the three time period strategy.

z5240149/ 60444.06870471329 36

References

[1] Philip Delff Andersen,Anne Iversen,Henrik Madsen,Carsten Rode, modeling of presence of occupants using inhomogeneous Markov chains. *Energy and Buildings*,69:213 - 223, 2014.

[2] Zhenghua Chen,Jinming Xu,Yeng Chai Soh, Modeling regular occupancy in commercial buildings using stochastic models, *Energy and Buildings*,103:216 - 223, 2015.

[3] Shide Salimi,Zheng Liu,Amin Hammad, Occupancy prediction model for open-plan offices using real-time location system and inhomogeneous Markov chain, *Energy and Buildings*,152:1 - 16, 2019.

[4] Wei Wang,Jiayu Chen,Xinyi Song, Modeling and predicting occupancy profile in office space with a Wi-Fi probe-based Dynamic Markov Time-Window Inference approach, *Energy and Buildings*,124:130 - 142, 2017.