Data Analyzing and Daily Activity Learning with Hidden Markov Model

GuoQing Yin and Dietmar Bruckner
Institute of Computer Technology
Vienna University of Technology, Austria, Europe
{yin, bruckner}@ict.tuwien.ac.at

Abstract—**To observe and analyze person's daily activities, and build the activities model is an important task in an intelligent environment.**

In an Ambient Assisted Living (AAL) project we get sensor data from a motion detector. At first we translate and reduce the raw data to state data. Secondly using hidden Markov model, forward algorithm, and Viterbi Algorithm to analyze the data and build the person's daily activity model.

Comparing individual observation with the build model to find out best and worst (abnormal) activities.

Keywords: Intelligent environment, hidden Markov model (HMM), forward algorithm, Viterbi algorithm.

I. INTRODUCTION

There are many papers about Markov chain and hidden Markov model: in paper [1] the author introduces the basic definition of Markov chain and the hidden Markov model, furthermore the applications of HMM. The authors in paper [2] describe a technique to learning the number of states and the topology of hidden Markov model from examples. Here the Bayesian posterior probability was used to choose the states for merging and for the stopping criterion. In paper [3] the author describes the EM Algorithm and parameter estimation for hidden Markov model. In paper [4] the authors give an introduction to the theory of hidden Markov model and illustrate how the hidden Markov model has been applied to problems in speech recognition. A factorial hidden Markov model is introduced in paper [5], that the method has advantage over unconstrained HMM in capturing data statistical structure. Hidden Markov model is used to allow for unequal and unknown evolutionary rates at different sites in molecular sequences in paper [6], here hidden Markov model allows for rates to differ between sites and for correlations between the rates of neighboring sites. In paper [7] for estimation the joint likelihood that between the observation data and the Markov state sequences a segmental k-means algorithm is used as objective function. In bioinformatics the author in paper [8] uses profile hidden Markov model to analyze the large-scale sequence in protein domains. Using the hidden Markov model to analyze the motion detector data, to learn the behavior of the user is expressed in paper [9] and [10].

A. AAL and ATTEND Project

Ambient Assisted Living (AAL) is a new search field, with the help from modern technology the life quality of the elderly improved and the independent living in their own home prolonged. But because of aging the elderly have their own problems, such as action obstacles, memory disorder … how can the elderly use the modern technology system?

The project ATTEND (AdapTive scenario recogniTion for Emergency and Need Detection) focus on developing a system that increases the independent living of elderly. In the living environment of user an intelligent, adaptive network of sensors are installed, in order to thoroughly observe the user's activities and behavior. In a time period the user's activities and behavior model will be learned by the system.

For example a user has activities at the living room. A motion detector installed at a corner of the living room and it records the daily activities. Then the activities model of the user will be build by hidden Markov model, forward algorithm, and Viterbi algorithm. In case of some abnormal activities happened, for example in the living room there are without activities in the morning – this kind of situation never happened before – so the system should according the build model to send an aware or alarm signal to user, neighbor, or care giver.

B. Basic Parameters

In the living room a motion detector installed, the motion detector sends the value "1" to the controller if there are activities from the person. Otherwise the motion detector sends the value "0" to the controller. For continuous activities or stillness the sensor value keeps on "1" or "0". This is the basic function from the motion detector.

Figure 1 from top to bottom shows the gathered sensor raw data (there are 694 values all), the filtered sensor value, and the histogram of the same data from one day. The x-axis is time. From the top of the figure we know that between 15 to 19 o'clock and close to 24 o'clock there are really quiet. At the other time there are activities but the activities density is different. The middle of the figure is the filtered sensor value from the raw data.

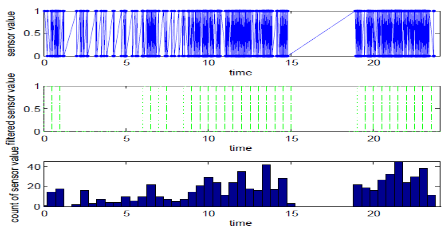

Fig 1. The data from one motion sensor in one day.

The filtering method is that at first segments the daily time into smaller time interval, for example 30 minutes, so there will be 48 time intervals in a day. Then the activities of the sensor summed together in each time interval. If the whole value bigger than a predefined threshold value, so this time interval will be treated as activities value “1”, others are value “0”. Here the predefined threshold value and the time interval play an important role.

From the middle of figure 1 we can see that between 1 to 6 o’clock there are without activities. It is not corresponding with the top of the figure 1. This is because of the “noise” value (for example the elderly rolls body in sleeping) be filtered. On the other hand if there are many activities such as between 0 and 1 o’clock (perhaps the user has sleeping disorder and difficult into sleeping), these activities will be translated to value “1” in the time interval.

In bottom of the figure 1 we use the histogram to comparison and validation the filtered sensor data with the original data. It is clearly that between 1 and 6 o’clock, and between 15 to 19 o’clock there are seldom activities than the other time interval. It corresponds with the middle of the figure 1.

If we gather some day’s data together we will get a general intuition of the user’s activity. In figure 2 there are 15 days filtered data. From figure 2 we know that between 0 to 5 o’clock the elderly has a few activities, between 5 to 10 there are a little more activities, from 10 to 15 there are most activities and after 15 the activities reduced and around 20 there are again more activities.

Till now we reduced the data account, send the value (activities) to each time interval. Using hidden Markov model, forward algorithm, and Viterbi algorithm we can get an activities model of the user.

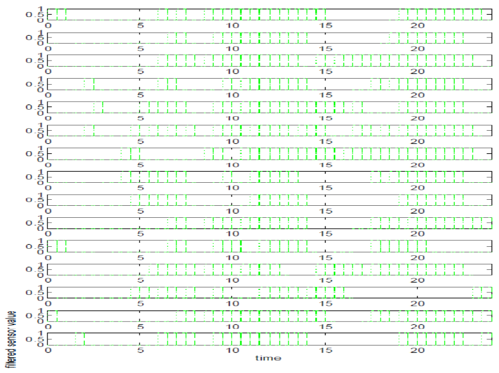

Fig 2. Filtered sensor value in 15 days

II. Markov Chain And Hidden Morkov Model

From above we get the activities from the user. If we chose $T_{interval}$ as 30 minutes, so there should be 48 values one day, for example $Q_{interval, ix} = \{ Q_{interval,1} = 0, Q_{interval,2} = 0, Q_{interval,3} = 1, \ldots, Q_{interval,\ m} = 0, \ldots, Q_{interval,46} = 1, Q_{interval,47} = 0, Q_{interval,48} = 0\}$. Here 1 <= m <= 48. If we treat each interval activity value (0 or 1) as “state”, so there should be 48 states each day.

A. Markov chain

There are some papers [1], [2] give the definition about the Markov chain. For us the discrete time first-order Markov chain is the most important. $P(Q_{t+1} = q_{t+1} \mid Q_t = q_t, Q_{t-1} = q_{t-1} \ldots Q_0 = q_0) = P(Q_{t+1} = q_{t+1} \mid Q_t = q_t)$. Here Q_t is a random variable from a countable state space at time t, q_t is the taken variable in a countable set at time t.

B. Hidden Markov model

The definition of HMM given in paper [1] and [2]. Briefly, an HMM consists of states and transitions like a Markov chain [2]. In the ATTEND project we connect hidden Markov model definition and the real application together. At first we build a hidden Markov model according the daily activities and then using the model to explain the observed sequence of daily activities. The first question we have to deal with is how to build a hidden Markov model. It needs some basic definitions and algorithm.

C. Basic parameters of hidden Markov model

There are some parameters characterize hidden Markov model, for a better understanding, on the following we will consult the real application to explain these parameters.

- The number of states N

Here we have 48 states in each day.

- The number of each state output distinct observation symbols M

Here we have output {0, 1}, so M = 2.

- The state transition probability distribution matrix A = $\{p_{ij}\}$.

$p_{ij} = p\ \{Q_{t+1} = j \mid Q_t = i\}$. (1)

$0 <= p_{ij} <= 1$ and $\sum_{j=1}^{N} p_{ij} = 1$, $1 <= i, j <= N$.

Here Q_t is the current state at time t.

- The state emission probability distribution matrix B = $\{b_{ik}\}$.

$b_{ik} = p\ (O_t = k \mid Q_t = i)$, $1 <= i <= N$, $0 <= k <= M$. (2)

Here O_t is the output symbol at time t.

- The initial state distribution $\pi = \{\pi_i\}$.

$\pi_i = p\ \{Q_o = i\}$. (3)

For example there are 2 initial states π_1 and π_2, π_1 include 4 values, all are “0” and π_2 include 6 values, all are “1”, so $\pi = \{\pi_1, \pi_2\} = \{0.4, 0.6\}$.

As we predefined above ($T_{interval}$ chosen as 30 minutes), so there should be 48 values one day. If we treat each value as a state there should be 48 states, but because of merging different states the states count will be reduced. There are two different situations that the states can be merged together.

The first situation is that merging the identical states:

For example in some time intervals in one day the activity value of the user keep on “1”, Q = {… 1, 1, 1, 1, 1,…}, because the state transition probability value between state t and state t+1 keep on 100% and the state emission probability value keep on the same, so these states can be merged in to one state. In the merged state there are 2 parameters: P (Q_{ii}) and P (Q_{ij}).

P (Q_{ii}) = N_{merged} / (N_{merged}+1) (4)

P (Q_{ij}) = 1 / (N_{merged}+1) (5)

Here P (Q_{ii}) is the “self-transition” probability, P (Q_{ij}) is the transition probability, and N_{merged} is the number that is merged states. P (Q_{ii}) + P (Q_{ij}) = 1. In this situation b_{ik} = 1.

The second situation is that merging the consecutive states and these states has different alternately states value:

For example in some time intervals on one day the activity value of the user is Q = {… 0, 1, 0, 1, 0, 1…}, the value “0” and “1” appear alternately. It has the 100% state transition probability value between state t and state t+1 and the emission transition probability with increased states count closer to 0.5. All these states could merge into one state. The parameters P (Q_{ii}) and P (Q_{ij}) computed as above but the emission transition probability is different.

b_{i0} = N_0 / N_{merged} (6)

b_{i1} = N_1 / N_{merged} (7)

Here N_0 is the count that all the states have value “0” and N_1 is the count that all the states have value “1”. It is clearly $N_0 + N_1 = N_{merged}$ and b_{i0} + b_{i1} =1.

We have discussed the transition probability distribution in the merging situation above. Another situation is the split situation: from time interval t to the next time interval t+1 there is more than one state connected with the same state Q_t. For example in state Q_t there are 10 values, all these values are “1”. In the next time interval t+1 there has a state that has 3 values and all the 3 values keeping “1” and another state has 7 values and all the 7 values are “0”. So the state transition probabilities are 0.3 and 0.7 separately.

III. THE FORWARD ALGORITHM AND THE VITERBI ALGORITHM

Given a hidden Markov model that means the parameter (π , A, B) are known, how we can find the probability of an observed sequence $Q^{(t)} = \{q_1, q_2,…, q_t\}$? Here each of the q is one of the observable set. The forward algorithm is used.

- Get the first transition probability a_1 for t = 1.

a_1 (j) = π (j) * b_{jt} (8)

Here j is the observation count of each observation set and $\sum\pi$ (j) = 1.

- For t >= 2 get the transition probability a_{t+1} (j)

a_{t+1} (j) = $\sum_{i=1}^{n}$ (a_t (i)*a_{ij})*b_{jt} (9)

- For t <= T repeat (9).

Here T is the length of the sequence.

Most of the time we are not only need the probability of an observed sequence but also need to find out the best interpretation of the observation. In this situation the Viterbi algorithm will be used. The essence point of the algorithm is to find the maximum value of each step.

- For t = 1

a_1 =argmax$_j$ (π (j) * b_{jt}) (10)

- For t >= 2

a_t =argmax$_j$ (a_{t-1} * b_{jt}) (11)

IV. RESULT AND CONCLUSION

A. Result

The basic data come from a motion detector installed in a living room. We gathered the data for 15 days and translate these data to 48 states (30 minutes as one state interval) for each day. Figure 2 is the activities value for each state in all 15 days. For avoiding the state sequences crossing of different days’ we sort these days at first and then merge the identical states and the states with alternately value from left to right. Figure 3 shows the result.

In figure 3 each small circle means a state with the activities value is “0” or “1”. From figure 2 we know that at the first time interval there are two different values “0” and “1” in all 15 days (on first day, eleventh day and fourteenth day there are activity value “1” at the first time interval). So we split the data at the first state, in figure 3 the last three states sequences indicated the daily sequences with initial interval value “1”, others are the daily sequences with initial value “0”.

Compare figure 3 with figure 2 it is clearly that the states count reduced.

Till now the hidden Markov model with parameter λ = (A, B, π) are build. With the model we can calculate the probability of the observation sequences, find out the best “explains” of the observation, with Viterbi algorithm to find out the best parameters in each step.

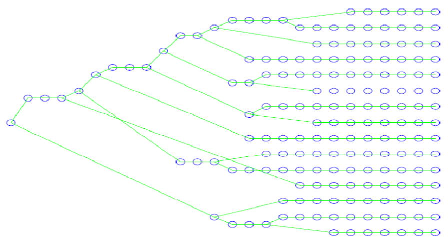
Fig 3. States after merging in days

Figure 4 shows the logarithm value for a chosen observation sequences in the 15 days. That means comparing a chosen day (from the 15 days) with each of the 15 days step by step. X-axis is the states for each day (or the sequences number on each day); y-axis is the logarithm value of the transition probability. In figure 4 the sequences with cyan circle is the best day that adapt to the chosen day. It has a logarithm value -19.56 after 48 states. In fact they are the same days. On the other hand the sequence with magenta star is the worst day that adapts to the observation sequences. It has a logarithm value -115.3 after 48 states. Because the parameters in A, B matrix all smaller (or equal to) than 1, so the logarithm value reduced step by step.

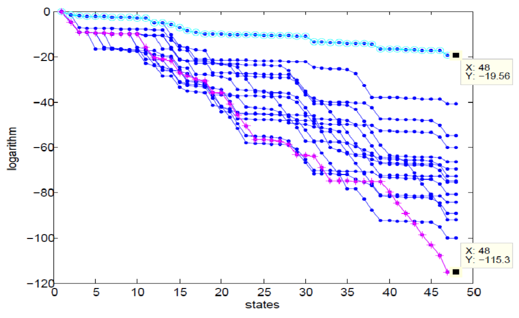

Fig 4. Compare the logarithm with chosen observation sequences in 15 days

It is often that in some steps there are without states value adapt to the observation sequence value, for example in step "s" the observation sequences has value "1", but in the same step the state has value "0", that means the state sequences cannot adapt to the observation sequences. In generally the state sequences should be defined "not adapt" to the observation sequences, and the comparing should be stop at once. The observation sequences should be comparing with other new state sequences. But in the real application what we meet is that: perhaps there just a few steps did not adapt to the observation sequences, and then the state sequences adapt to the observation sequences again. So in such situation we send a probability 1% to the transition matrix A in the step, to allow the comparing continues on. In figure 4 the logarithm value with steep drop connections are this kind of situations.

Figure 5 shows the logarithm value for a random observation sequences comparing with the 15 days. Because it is a random observation sequences there are many steps not adapt to the hidden Markov model, so the logarithm value reduced generally. The result is the best sequences adapt to the observation sequences has a logarithm value -129.1 and the worst logarithm value is -170.7.

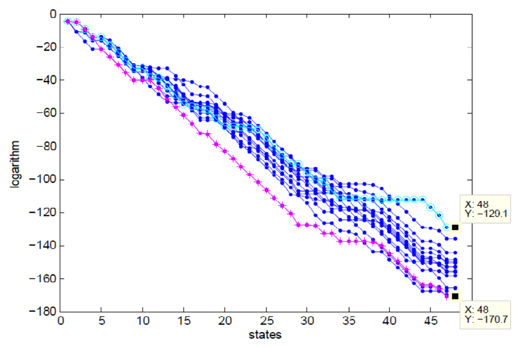

Fig 5. Compare the logarithm with random observation sequences

From figure 5 we know that in the best adapt sequences (the sequences with cyan circle) there are not each step has the maximum logarithm value. Using the Viterbi algorithm we can get the best sequences that adapts to the observation sequences. The result showed in figure 6. In figure 6 the sequences with green triangle is the best sequences adapt to the observation sequences. It comes from different states sequences.

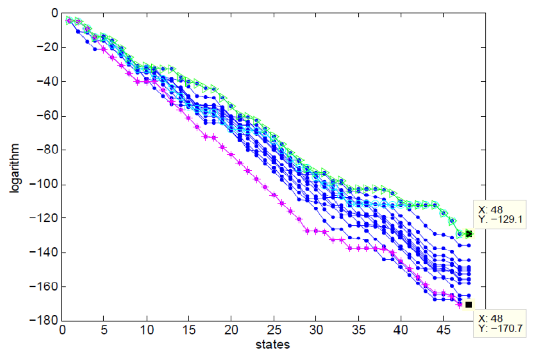

Fig 6. Compare the logarithm with random observation sequences and find the best value in each state

From above we explain the hidden Markov model, forward algorithm, and the Viterbi algorithm with real data. The next question is how we can find out the "best standard" daily sequences? That means, for example, we should find out a day from the 15 days that can typically represent the living activity model of the user.

We have sorted the 15 days (in figure 2) according the activities value and the new order is: 2, 10, 3, 12, 13, 9, 8, 7, 5, 4, 6, 15, 14, 11, and 1. Now we according the new order compare each day with the whole 15 days. The result shows in figure 7.

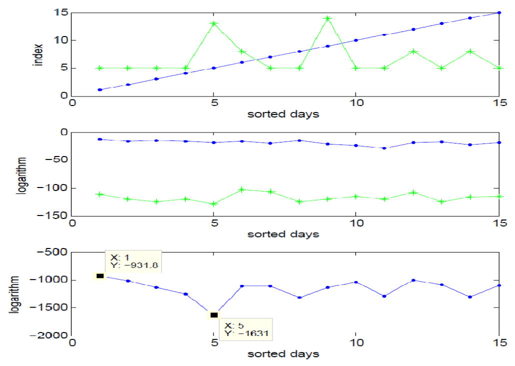

Fig 7. Compare each day to all the 15 days

The top of figure 7 is the index that which days are the best and the worst day to the compare day. X-axis is the sorted days with new order from 1 to 15. Y-axis is the index according the new order. The blue points and the line are the best corresponding day's indexes that in keeping with the comparing days. The green star and line are the worst corresponding day's indexes. It is clearly that the same day compare with itself will get the best result, so the blue points are connected to a line. Contrary the green points are different, the points connection are not a line and especially the day's index number "5" (it indicated the 13th day in figure 2) is mostly worst day to each comparing days. If we collate the index number "5" to the order, so we could find that the thirteenth day of the figure 2 is the worst day of all the 15 days. But this is just a rough judgement, we need proof with parameters.

The middle of figure 7 is the logarithm value for the comparing. The best day's value focus on -15 to -30 and the worst day's value concentrates in -100 to -130.

The bottom of the figure 7 is the sum of logarithm value from each comparing. For example a chosen day compare with all the 15 days and there should be 15 values, these values indicated the divation between the chosen day to all the 15 days. Then add all the 15 value together, the sum directly denotes the divation between the chosen day to all the other days. In such a way we can get the best standard day and the worst day in all the days.

From the bottom of the figure 7 we see that the day with index "1" is the best day with logarithm value -931.8 and the day with index "5" is the worst day with logarithm value -1631. Through collating the index order we will find that the 2th day of the figure 2 is the best day and the 13th day of the figure 2 is the worst day. Again the "best day" means a day has the maximum likelihood to all the 15 days and the "worst day" is a day has the minimum likelihood to all the 15 days.

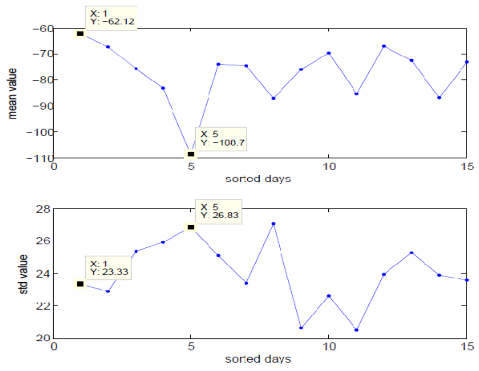

Fig 8. Mean and std logarithm value all the 15 days

Figure 8 shows the mean and std (standard divation) value of all the 15 days. From the top of the figure we can see that the day with index "1" has a mean vaule -62.12 and the day with index "5" has a value -108.7. It indicats again that the day with index "1" has a better similarity to all the 15 days but the day with index "5" is more far away. The bottom of the figure shows that the day with index "1" has a std value 23.33 and the the day with index "5" has a std value 26.83. It illustrates again that the day with index "1" has more similarity to all the 15 days than the day with index "5".

Furthermore according the deviation of logarithm value between the "best standard" sequence and the observation sequence we can judge if the observation sequences normal or abnormal. For example in figure 8 we get the best logarithm mean value -62.12 as a standard value, and think of the standard deviation value 23.33, so there should be a logarithm value range from -38.79 to -85.45. Then comparing the observation sequences to the standard sequences, if the logarithm value in the range, so it is a normal sequences, else the observation sequences is abnormal sequences. It indicates an abnormal daily activity.

B. Conclusion

Hidden Markov model, the forward and Viterbi algorithm is a powerful tool for unsupervised learning and very useful in real-world applications.

For the application of AAL the stable life style of the user is the basic of a useful learning result.

V. OUTLOOK

In the future some more robust learning algorithms should be developed and tested. For example backward algorithm will be used together with forward algorithm to reduce the states further more.

REFERENCES

[1] Jeff Bilmes (2002). "What HMMs Can do", UWEE Technical Report, Number UWEETR-2002-0003, January 2002.

[2] Andreas Stolcke and Stephen Omohundro, "Hidden Markov Model Induction by Bayesian Model Merging", Berkeley CA 94704, advances in Neural Information Processing System5, San Mateo, CA, Morgan Kaufman, 1993.

[3] Jeff A. Bilmes, "A Gentle Tutorial of the EM Algorithm and its application to Parameter Estimation for Gaussian Mixture and hidden Markov Models", international computer science institute, Berkeley CA, 94704, April 1998.

[4] L. R. Rabiner and B. H. Juang, "An Introduction to Hidden Markov Models", IEEE ASSP MAGAZINE, JANUARY 1986.

[5] ZOUBIN GHAHRAMANI and MICHAEL I. JORDAN,"Factorial Hidden Markov Models", Machine Learning, 29, 245-273 (1997).

[6] Joseph Felsenstein and Gary A. Churchill, "A Hidden Markov Model Approach to Variation Among Sites in Rate of Evolution", 1996 by the Society for Molecular Biology and Evolution. ISSN: 0737-4038.

[7] BIING-HWANG JUANG and L. R. RABINER, "The Segmental K-Means Algorithm for Estimating Parameters of Hidden Markov Models", IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SINGAL PROCESSING. VOL 38, NO 9, SEPTEMBER 1990.

[8] Sean R. Eddy, "Profile hidden Markov models", Bioinformatics review, Vol. 14 no. 9 1998, Pages 755-763.

[9] Dietmar Bruckner, Brian Sallans, and Gerhard Russ, "Probability Construction of Symbols in Building Automation Systems", in: Proceedings of 2006 IEEE International Conference of Industrial Informatics INDIN' 06, S. 6, Singapore, 2006.

[10] Dietmar Bruckner, Brian Sallans, and Roland Lang, "Behavior Learning via State Chains from Motion Detector Sensors", Biometrics '07 December 10-13, 2007, Budapest, Hungary.