Hidden Markov Model for Censored Production Rules

Nabil M. Hewahi
The Islamic University, Gaza, Palestine
nhewahi@iugaza.edu.ps

ABSTRACT

In this paper, we use hidden Markov model which is based on statistical model as a higher knowledge representation scheme to induce Censored Production Rules that are well known in real time systems. We present a modified version of censored production rule that can fit with hidden Markov model and present a scheme to compute the certainty values of the obtained conclusions out of the induced rules. To compute the certainty values for the rule actions (conclusions), we use only the probability values associated with the hidden Markov model, and there is no need to use any of the other well known certainty computation approaches.

Key words: hidden Markov model, censored production rules, certainty factor

1.Introduction

In the introduction, we present an overview of hidden Markov model and the censored production rules to make a simple pavement for the research objectives in the next section.

1.1 What is Hidden Markov Model?

Hidden Markov Model (HMM) is a statistical model based on probabilities used in various applications such as cryptanalysis, machine translation, speech and hand recognition, natural language processing, gene prediction and bioinformatics [1][5][17-22]. A Markov model is a probabilistic process over a finite set, {S1, ..., Sk}, usually called its states. Each state-transition generates a character from the alphabet of the process. In a regular Markov model, the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. In a hidden Markov model, the state is not directly visible, but variables influenced by the state are visible. Each state has a probability distribution over the possible output tokens. Therefore, the sequence of tokens generated by an HMM gives some information about the sequence of states. There are three problems associated with HMM, evaluation, decoding, and learning problems. The evaluation problem, given the parameters of the model, compute the probability of a particular output sequence, and the probabilities of the hidden state values given that output sequence. The evaluation problem can be solved by forward-backward algorithm. The decoding problem, given the parameters of the model, find the most likely sequence of hidden states that could have generated a given output sequence. This is solved by Viterbi algorithm The learning problem, given an output sequence or a set of such sequences, find the most likely set of state transition and output probabilities. This problem is solved by the Baum-Welch algorithm [23].

1.2 Censored Production Rules

The standard rule structure is very well known in the area of expert systems and also currently in data mining clustering and prediction. The structure of standard rule is (<IF condition THEN action>). As an extension of standard production rule, Michalski and Winston proposed the

censored production rule (CPR) of the form (<IF condition THEN action UNLESS censor>) as an underlying representational and computational mechanism to enable logic based systems to exhibit variable precision logic (VPL) in which certainty varies, while specificity stays constant [16]. CPRs are used in real time applications. The more time we have, we check more censors and become more certain about our conclusions. If we do not have enough time to check the censors (or some of them), the system takes the action of the rule with less certainty. The form of CPR is $P \longrightarrow D \lfloor C$, where P is the premise, D is the decision, and C is the censor. The premise is a conjunction of literals; the decision is a single literal; and the censor is a disjunction of literal. CPRs embody both object level and control level information. The object definition is false most of the time; we have certain expectations concerning the character of inferences made with such rules. These expectations may be used to control the inferences. To understand the implication of CPR, Michalski and Winston presented a quantitative definition for it, where two parameter γ and δ have been introduced. A CPR is then written: P D $(C1 \vee UNK) : \gamma , \delta$,
where γ = prob(D\P), certainty of P D, when it is not known whether $(C1 \vee UNK)$ holds (UNK means not yet known censor conditions). The implication $P \longrightarrow D$ is certainty 1, when $(C1 \vee UNK)$ is known to be false. When δ = prob(D\P& $\neg$ C1), it is the certainty that $P \longrightarrow D$, when C1 is true. Obviously the a priori certainty of $\neg (C1 \vee UNK)$ must be equal to or smaller than a priori certainty that $\neg$ UNK. Therefore, $\gamma \leq \delta$. Note that $\gamma = 1$ if it is certain that there are no conditions in the censor other than C1. As an example to CPR.

IF Working_Day $\longrightarrow$ John_in_office
$\lfloor$ John_is_sick, John_on_leave

The system checks first working-day, if it is true and we have more time, the system checks the censors john-is-sick and john-on-leave based on the given time.

A lot of work and applications have been done on VPL systems [2][3][4][6-15].

2. The Research Objectives

There are two main objectives of this research:

a) How to use HMM to represent CPRs.
b) How to compute the certainty factor for the achieved conclusions.

Before explaining our objectives, we shall raise a question, why should we try to make a relation between HMM and rule based systems in general ? As we know HMM is used in various AI applications, those applications might use various knowledge representation schemes based on the nature of application and the knowledge structure. One of the very well known knowledge representations is rule based structure. In rule based systems, we usually use certainty factor to scale the certainty of the rule, which can be computed with many techniques and approaches such as Baysian approach, emycin approach and dempster-shafer approach. This raises another question, can we use the probability values used with HMM as certainty values for the obtained rules and how this can be done. To make the idea clear, we are trying to make HMM as a knowledge representation plus a model for fixing the certainty factor of the induced rules from the HMM without using any additional approach for computing the certainty factors.

In the first objective, we shall try to find a method to represent and induce CPRs out of a given HMM. The problem in this case is how to specify the rule censors and how to write the CPR.

The second goal is how to utilize the probability values of the HMM to get the certainty of the obtained conclusions from the CPRs.

3. The Proposed Approach

In this section, we discuss our approach to tackle our two defined objectives.

3.1 Representation of CPR using HMM (First objective)

In this section, we concentrate on the first objective. Let us assume the figure shown in Figure 1.

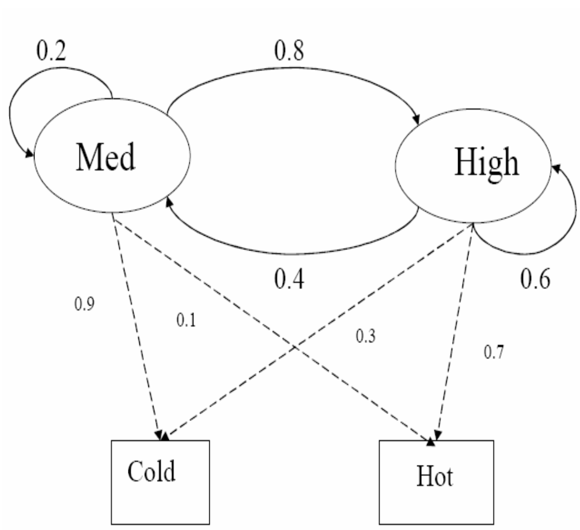

Figure 1. A HMM to describe a relation between the states Med. and High with the observations (invisible states) cold and hot.

The summary of figure 1 says:

P(cold | med) = 0.9, P(hot | med) = 0.1, P(cold | high) = 0.3, P(hot | high) = 0.7, P(med |med) = 0.2, P(high | med) = 0.8, P(high | high) = 0.6, P(med | high) = 0.4

Before trying to formulate a way to put the Figure 1 in a CPR, we try first to present the figure using a rule structure as shown below:

If med Then cold 0.9
If med Then hot 0.1
If high then cold 0.3
If high then hot 0.7
If med then med 0.2
If med then high 0.8
If high then high 0.6
If high then med 0.4

Now it is to be noted that the relation between the state med and the observations hot and cold are divided to values 0.9 and 0.1 which means the observation hot is occurring more likely than cold given med. This may allow us to make a a modified CPR (MCPR) in the following form

If state then observation/state (p1) unless c_1 then observation/state (p2)

Where p1 and p2 are the probability of observation/state given state, and c_1 is the censor condition. To make the format easy to understand, let us consider the following MCPR

If med then cold (0.9) unless c_1 then hot (0.1)

This rule is an improvement of the ordinary CPR, it says if med is true then it is cold unless c_1 is true (c_1 has to be false in case we have time to check it). If c_1 is true then the result would be hot. In the ordinary CPR we decide about the conclusion if the condition is proved to be true but we do not know the result if c_1 is true. Other rules that we can form are

If high then hot (0.7) unless c_2 then cold (0.3)
If med then high (0.8) unless c_3 then med (0.2)
If high then high (0.6) unless c_4 then med (0.4)

The main question here, how can we represent the censor c associated with the rule ? For this purpose we have to present each censor/s related to a certain observation above the observation's arrow with the less value of probability. To understand this, let us look at Figure 2.

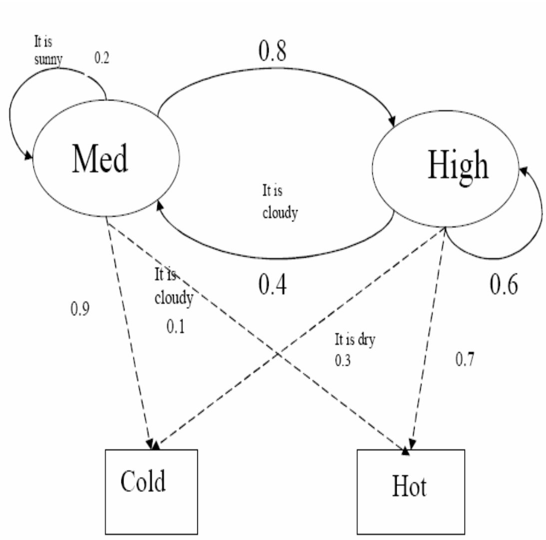

Figure 2. A HMM to describe a relation between the states Med. and High with the observations (invisible states) cold and hot. Also the censors concerned with alternative observation and states are presented on their corresponding arrows.

Based on Figure 2, we can write the rules as below:

If med then cold unless It is cloudy then hot
If high then hot unless It is dry then cold
If med then high unless It is sunny then med
If high then high unless It is cloudy then med

Now the rule one reads as, if it is med is true then it is cold unless it is cloudy then the result would be hot. In general we might have more than one censor condition in the rule such as

If condition then action1 unless c_1, c_2,… c_n then action2

We have first to remember that censors are usually not checked unless we have time to do that. Also, to get advantage of the time, we have to arrange the censors in some way that can make us get benefit in terms of time out of this arrangement. In the above rule structure, we deal with the censors in a way that each is a separate unit, which means the occurrence of any of the censors can lead to a failure of taking action1. This can be represented above the arrows connecting the visible states with the observations as c_1, c_2, c_3. In case we have time to check the censors, c_1, c_2 and c_3, each has to be false to take a decision of action1. In case any of the censors is true, no need to check the rest of the censors even if we have enough time, and action 2 is taken. In this way, it is good to have the censors ordered based on their occurrence (priority / importance), which will allow us to check the most important censors as the time permit. This will give a better certainty and also supports the real time systems. We shall discuss this point in the next section.

3.2. Certainty Factor Computations (Second objective)

Let us now go for more complex computation:

If sunny then walk (0.8) unless sick , has-guest then at-home (0.2)

If we have enough time, we can check the censors sick and has-guest and if they are true, we can say that we are 100% that the result is true in this case. We are trying to elaborate the probability values and work with them as certainty factor and we do not want to use other approach. In this case we shall assume that 0.8 is our certainty that walking occurs in case it is sunny, now if we check sick censor and find out that it is false, we go forward and check the next censor and if it is also false then the 0.2 is added with the 0.8, and we say that we are certain from our conclusion with value 1. If sick censor is true, then we do not need to check more censors and take a decision with at-home with certainty 1.0. The following algorithm summarizes the computation of the certainty factor of the rules.

1. Choose the rule to be fired from the matching box based on the used control strategy.
2. Divide the p2 value equally among the censors (for now, our assumption is that all the censors are of the same importance). Let us call the censor share of the p2 for the ith censor , shr-c_i
3. Check the condition part of the rule, if it is true do the following
   cf = p1
   - while time is not over do
     check the censor i based on its order in the rule
     if the censor is false then
       cf = cf + por-c_i
     if the censor i is true then
       cf = 1.0
       take decision of action2
       exit the while
4. if time is over, then take a decision of action1 (we may quit of while because one of the censors is true).
5. if the condition part is false , then check next rules based on the used control strategy.

To make the above algorithm clear, let us consider the same rule presented above, in this case the sick censor will have a share of 0.1 and a share of 0.1 for has-guest censor. Let us assume that it is sunny and we have time, we check sick censor and find it is false, we add 0.1 to 0.8. Now we check again for the time, and let us assum that there is no more time, so we take a decision of walking with certainty 0.9. If still we have time, we check has-guest censor condition, and if it is false, we take a decision of walking with 1.0. In contrary, if we check the main condition and we still have some time, we check sick censor. If sick is true, then we take a decision at-home with certainty factor 1.0 and we stop. In this case we do not need to check for other censors even if we have time to do so.
In case the censor conditions are sorted according to their importance, then we should not deal with them equally. To make the idea clear, let us consider the following MCPR

If condition then action1 (0.7) unless c_1, c_2,… c_n then action2 (0.3), given that c_1 is having higher priority than c_2, and c_2 is having higher priority than c_3 and so on. We computer each probability portion for censor i as below

$$\text{shr-}c_i = (n - i + 1) / \left(\left(\sum_{i=1}^{n} i \right) * \ p2 \right),$$

where n, is the number of censor conditions. To make this formula clear, let us consider the following example

If cond1 then action1 0.8 unless c_1, c_2 , c_3 then action2 0.2

Then shr-c_1 = (3/6) * 0.2 = 0.1
shr-c_2 = (2/6) * 0.2 = 0.06666
shr-c_3 = (1/6) * 0.2 = 0.03333

Using this technique, after checking the cond1 and having more time, we check the censor c_1. If c_1 is false and no enough time to continue for checking, The system takes a decision action1 with certainty 0.8 + 0.1 = 0.9. If more time is available to check one more censor, the system checks c_2. if c_2 is again false then the system becomes certain of action1 by 0.8+0.1+ 0.06666 = 0.96666

4. Complex MCPRs

One big question may arise, if there are many observations associated with a state, how can we construct the MCPR and how can we do the computations. To illustrate the case, let us consider the following situation:

'Rainy' : {'walk': 0.1, 'shop': 0.4, 'clean': 0.5},
'Sunny' : {'walk': 0.6, 'shop': 0.3, 'clean': 0.1},

where rainy and sunny are the visible states and walk, shopping and cleaning are the observations. We can construct the following MCPR as below

If rainy then cleaning (0.5) unless c1 then shopping (0.4) unless c11 then walking (0.1)

In this case it is important for taking a decision of cleaning to prove the falsity of c1 and c11 if there is enough time to check them. If we have time to check one censor, we can check c1. If it is false we can take a decision of cleaning with 0.9. If c1 is true, then the system will take a decision of shopping with a certainty of 1.0. If time is enough to check the censors c1 and c11 and they both proved to be false, then action 1 will be achieved with certainty 1.0. In case c1 is false and c11 is true then a decision of walking is achieved with certainty 1.0.

5. Conclusions

In this paper we used the HMM as a knowledge representation to induce rules that are very useful in real time systems, this rule is called CPR. We also presented a new form of CPR and called it MCPR. The main difference between the two structures is that using MCPR gives the ability to know what action to be taken if the censor condition of the rule is true, whereas in CPR there is no way to know that. Probability values associated with HMM are used to compute the certainty values of the concluded actions of the MCPR. This is an important step, because we need not to make another overhead by using some other certainty factor computing approaches. During computations, censor conditions can be treated equally or based on their priority. Some of the future directions could be doing some implementations related to MCPRs and applying such systems to real time applications. Another direction would be developing a software that can take Markov model as input and produces MCPRS as output. This will reduce a lot of work in developing real time systems based on MCPR.

References

[1] L. Allison, L. Stern, T. Edgoose and T.Dix, "Sequence Complexity for Biological Sequence Analysis", Computers and Chemistry, 24(1), Jan. 2000, pp. 43 - 55.

[2] K.Bharadwaj, N. Hewahi, M.Brando, "Adaptive hierarchical censored production rule-based system: a genetic algorithm approach, in: S.L.Borges, C.A.Kaestner(Eds.), Lecture Notes in Artificial Intelligence, Advances in Artificial Intelligence, the Brazilian Symposium on Artificial Intelligence SBIA'96, vol. 1159, Springer, Berlin, 1996, pp. 81 - 90.

[3] K.Bharadwaj, N.Jain, "Hierarchical censored production rules (HCPR) system, Data and Knowledge Engineering, vol. 8, North Holland, Amsterdam, 1992, pp. 19 - 34.

[4] K.Bharadwaj, Neeraj, C.Goe, "Hierarchical censored production rules system employing Dempster-Shafer

uncertainty calculus, Information and software Technology 37(8), 1995, pp.453-460.

[5] V. Fonzo, F. Aluffi-Pentini and V. Parisi, “Hidden Markov Models in Bioinformatics”, Current Bioinformatics, 2, 2007, pp. 49-61.

[6] P.Haddway, “Implementation and experiments with VPL inference system”, Intelligent System Group, Department of Computer Science, University of Illinois, 1989.

[7] N.Hewahi, “Credit Assignment Scheme for the General Rule Structure (GRS)”, Proceedings of the 1st Indian International Conference on AI, Hyderabad, 18-20 December, 2003, pp 883-842.

[8] N.Hewahi, “Principles on Integrating General Rule Structure(GRS) Based Systems and Neural Networks”, Proceedings of the 2004 International Conference on Artificial Intelligence (IC-AI’04), Las Vegas, Nevada, USA, June 21-24, 2004.

[9] N. Hewahi, “Censored Production Rules to General Rule Structure Converter” ,Journal of WSEAS Transactions of systems, issue 5,vol.3, 2004, pp 2074-2079.

[10] N. Hewahi, " Training and Extracting Real Time system Symbolic Rules Using Neural Networks", Asian Journal of Information Technology, Vol. 4, no.10,2005, pp 920-926,2005.

[11] N. Hewahi, "Extraction Classification Variable Precision Rules From Induced Decision-Exception Trees", Proceedings of the 4th Inter. Conference of Information Technology in Asia(CITA'05), December 13-15, Sarawak, Malysia,2005.

[12] N.Hewahi, “Real time variable precision logic”, in: F.M.de Oliveira(Ed.), Lecture Notes in Artificial Intelligence, vol. 1515, Springer, Berline, 1998, pp. 201-208.

[13] N.Hewahi, “Real time variable certainty systems”, Proceedings of the Fourth World Congress on Expert System, Mexico 4-8 March, 1998, pp. 692-697.

[14] N. Hewahi, “A general rule structure”, Information and Software Technology, 44, 2002, pp. 451-457.

[15] N. Hewahi, K.Bharadwaj,” Bucket brigade algorithm for hierarchical censored production rule-based system”, International Journal of Intelligent Systems”, 11, 1996, pp.197-225.

[16] R. Michalski, P. Winston, “Variable precision logic”, Artificial Intelligence, 29, 1986, pp. 121-145.

[17] J. Li, A. Najmi, R. M. Gray, Image classification by a two dimensional hidden Markov model, IEEE Transactions on Signal Processing, 48(2), February, 2000, pp. 517-33.

[18] Lior Pachter and Bernd Sturmfels. "Algebraic Statistics for Computational Biology". Cambridge University Press, 2005. ISBN 0-521-85700-7.

[19] B. Pardo and W. Birmingham “Modeling form for on-line following of musical performances”, AAAI-05 Proc., July 2005.

[20] L. Rabiner, “A tutorial on hidden Markov modeland selected applications on speech recognition”,

Proceedings of IEEE, vol.77, no.2, 1989.

[21] K. Seymore, A. McCallum, and R. Rosenfeld, “Learning hidden Markov model structure for information extraction”, AAAI 99 Workshop on Machine Learning for Information Extraction, 1999.

[22] L. Stern, L. Allison, R. Coppel, and T. Dix, “Discovering patterns in Plasmodium falciparum genomic DNA”, Molecular and Biochemical Parasitology, 118(2), 2001, pp.175-186.

[23] http://en.wikipedia.org/wiki/Markov-model