PERFORMANCE EVALUATION OF PULSED RADAR IDENTIFICATION USING HIDDEN MARKOV MODEL

HOSSAM E. ABOU-BAKR HASSAN*, AHMED EL-MAHDY*
AND MOHAMED ASAAD ABD-ELRAZEK*

ABSTRACT

Due to the increasing complexity in modern radar systems, it has become apparent that new ways of classify received signals are required. In this paper, the performance of a proposed method for radar system identification using hidden Markov model (HMM) is evaluated. In this method, a given radar system is modeled as a finite state automaton. In so, doing, it is possible to uncover the underlying system processes in a probabilistic fashion using hidden Markov models. Artificial deterministic signals are used to show that HMMs can provide adequate signal recognition. It will be shown that a perfect exists that is directly related to the number of symbols per period. A slightly adjusted version of this perfect model provides low false recognition rates in a threat library consisting of three models so long as the observation errors do not become too frequent. The simulation results show that if the observation errors are introduced, then, it becomes necessary to retrain the HMM with an error corrupted version of the original training sequence in order to improve the model's robustness.

KEY WORDS

Signal processing, Radar, Hidden Markov Model.

* Egyptian Armed Forces

1. INTRODUCTION

The growth complexity of modern and indeed, future radar design is required a new approach to radar system identification for Electronic warfare (EW) purposes. Conventional assumption about the nature of radar signals are provided to be less and less valid; namely that the signal can be assumed to be stable and predictable. Simple cross-correlation of the received signal with a library of known signals is in adequate in some applications when the signal source transmits pulses in a pseudo-random fashion. Instead, if the source is assumed to be a finite automaton then it is possible to model it using a hidden Markov model (HMM). HMMs have been used extensively in many applications requiring a signal model for practical functions such as classification, decoding and prediction [1]. Much of the theory used in this paper has been based on the knowledge gained about HMMs for speech recognition problems and could be appropriately applied to radar system identification. HMMs have been used in a host of pattern recognition application such as: inferring grammar of simple language, relation ships between pairs of DNA sequences, signature and face recognition [3,4].

Due to the fact that modeling a modern radar system as a cycle stationary source is no longer justified [1], previous methods of signal classification such as histograming of pulse arrival times and cross-correlation of received signals with known ones will prove to be unreliable and ineffective. Two key advantages to modeling a radar as a finite state automaton are: (1) finite state automaton are flexible and can be designed so as to capture all the required temporal information about dynamic systems; (2) even though the input symbols controlling state transitions in the state machine are hidden in most EW applications, they can be uncovered probabilistically using HMMs.

Radar system identification uses a pattern recognition as in a field that has successfully used HMM in many applications over last twenty years [2]. They are used to try to replicate simple (first-order) Markov process whose state (and thus symbol outputs) at time t (the present) depends only on the last output at time t-1 [2]. Although the actual signal behind modeling may not originate from a true first order Markov source (that is, the present output symbol may depend on more than just the previous output), the success of HMMs have enjoyed in classification applications over the years is evidence to the fact that the simple Markov source model is usually a valid one.

The information used in this paper to characterize a given radar signal Is that of the PRI where a pulse is represented by a '1' while a '0' signifies no received pulse. it is assumed that any received signal has been put through a front-end-signal processing block so that requisite functions such as sampling and deinterleaving [7] have been accomplished. There are four main problems using HMMs as described in [1] of which three are addressed in this paper, these are: (1) Classification: correctly choosing he model out of a library that best represents the received signal source; (2) Decoding: having chosen a particular model, what state sequence best describes the observed sequence? (3) Predicates:2given a model and partial observation

sequence, what is the next symbol ('0' or '1') most likely to be? (4) Training: given an observation sequence, what are the optimal parameters for the HMM so that if it is able to reproduce the sequence?

This paper will focus mainly on the training and classification problem, but will briefly touch on the decoding issue and discussing its implications for radar system identification. Training is by far the most important and involved problem because the parameter chosen to represent the received waveform necessarily dedicate the performance of the other three problems. That is to say, if there is poor training a non-optimal choice for the model's parameters, then the probability of proper classification degrades and prediction decreases substantially. The principle objective of this paper is to provide a method for choosing the optimal model parameters for an actual radar signal so that a library of HMMs can be created and used for practical EW tasks. Although the only data used in the simulations to characterize a radar signal is the PRI, the result learned from this paper represents a stepping-store for further inclusion of many the other well-known signal parameters such as carrier frequency, polarization and pulse duration. As a result the recognition performance can only increase with the addition of this information. This paper is organized as follows. In section 2, the training of a HMM will first be studied when there are no errors in the sequences and its performance is tested as a function of the number of states in the model. In section 3, the recognition performance will then be studied when errors are intentionally added into an observation sequence at a known rate. In section 4, a brief insight into the use of a uniform initial state vector will be given. Section 5 will summarize the main conclusions.

2. NUMBER OF STATES

The choice of number of states to be used in a HMM has a fundamental influence on its recognition performance. In isolated word, in recognition there are two major ways of tackling this issue. The first idea is to have the number of states roughly equal to the number of sounds or phonemes within a given word. The second is to let the number of states be approximately equal to the number of observations in a spoken version of the word [2]. The former uses a so-called left-right model which faces the state transitions to either loop back onto the same state to proceed to the next state, but never return to the previous state. Radar system identification (RSI), on the other hand, is interested in creating a model for an entire dynamic radar system. That is, the HMM must be able to recognize every possible word and combination there of that the radar is capable of producing. To do this, it is conceivable that we would create a separate HMM for each word from a particular radar and then use a state transition matrix whose elements dedicate the transition probabilities from one HMM to another. Another approach would be to consolidate all the information from the radar signal into one global HMM that is not left-right, but would allow for any possible transition. Figure 1 shows a 4-state left-right model[1]. Note that the model must begin in state 1, and will always end in state 4.

In the Figure 2 shows a fully connected or ergodic model for a radar capable of

[1] some of the examples have been taken from Rabiner's ([2],p.266). Rabiner defines an ergodic model as one in which every $a_{ij}>0$. The author has used the word “global” in order to differentiate between a finite state network where notes represent actual HMMs of words or phonemes ([2],p.283) instead of just states.

producing three words. The letters A, B and C represents the actual HMM that produce each of the three words. These sub-HMMs could either be left-right (Figure1) fully ergodic or partially ergodic. Lastly, we have a partially ergodic global model in Figure3.
Isolated word recognition prefers to use the left-right model in Figure 1 because time can be readily associated with state progression. That is to say, time can only go forward so the model can not allow for transition to treat back to previous state [2]. Connected or multiple word recognition has successfully employed the model shown in Figure 2 where the transitions from one word model to the next are derived using either a level building approach (similar to Figure 2) or a time synchronous Viterbi search [2].

Although Figure 2 could be a promising way to proceed for modeling of radar signals (deterministic or otherwise), the global partial ergodic model was chosen as the best way to proceed due to its simplicity. That is, if all information of the source can be retrieved from a single HMM, why try to model the same source using several HMMs? The question thus arises, what physical meaning to the states have in this case since, in the left-right model, they could either represent the number of phonemes per word, or the number of observation per word. The answer is simply that the states do not have a physical meaning, but are simply information bins that we use to model the source.

Consider that we have the following deterministic periodic training sequence 0010100101… which is known to contain no errors. The question thus arises, what would be the best choice for the number of states in terms of the training probability $P(O|\lambda)$? Figure 4 shows the training probability versus number of states for different length of this sequence, it can be seen that, regardless of the training sequence length, at 5 states and its doubles there is a perfect training probability (log probability equal to 0) while any thing else the log probability less than zero. The fewer the number of states, the smaller the training probability, with longer test sequence performing worse than short ones.

It would appear that something special happens at five states in that all the information about the signal has been captured at this point due to the perfect training probability. If we look at the training sequence it is clear that it is composed of repeating units of the sequence 00101 which is five symbols long. The ***A, B*** and π parameters for a typical model with five states are:

$$A=\begin{bmatrix}0&1&0&0&0\\0&0&1&0&0\\0&0&0&1&0\\0&0&0&0&1\\1&0&0&0&0\end{bmatrix},\ B=\begin{bmatrix}1&1&0&1&0\\0&0&1&0&1\end{bmatrix},\pi=\begin{bmatrix}1&0&0&0&0\end{bmatrix} \quad (1)$$

We will introduce the concept of the perfect model as any HMM that is capable of reproducing a given training sequence with probability 1. For this training sequence perfect models can occur for $N=5$ states. It should be clear that, in terms of recognition as long as the observation sequence is synchronized to the model (begins with 00101…) then recognition probability will be the same as the training probability for a given number of states. That is, a given model with training probability P will classify the same error-free and synchronized observation sequence with probability P assuming that the model had been sufficiently trained.

Assuming we have sufficiently trained a model it is true that the perfect model for any deterministic sequence must contain at least as many states as the number of the symbols per period. Figure 5 shows the training probability of the sequence 000100100010010001001……. Once again, a log probability of zero occurs at N=7 which corresponds to the seven symbols in each period.

A cursory explanation of this observation can be seen through the way in which the forward-backward procedure computes the training probability.

$$\alpha_1(i)=p\left(o_1,q_1=S_i\middle|\lambda^{(L)}\right)=\pi_i b_i(o_1) \qquad 1\le i\le N \quad (2)$$

$$\alpha_{t+1}(j)=\sum_{i=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1}) \quad (3)$$
$$1\le j\le N \ , \ 1\le t\le T-1$$

And that the sum of the terminal forward variables over all states in the model is defined as the probability of producing the given sequence ***O*** given the model λ.

$$p(O|\lambda)=\sum_{i=1}^{N}\alpha_T(i) \quad (4)$$

If there are any values in either ***A*** or ***B*** are on the interval (0,1) then it is clean that the forward variable $\alpha_t(i)$ will also be less than 1 due to the multiplication in equation (3). The longer the sequence the smaller the overall probability. This can be seen in Figure 4 where $P(O|\lambda)$ for the model trained on the 20-period sequence is much smaller than that of the 4-period sequence.

The only way for ***A*** and ***B*** to contain only 1's and 0's is for the number of states in the model equal to the number of symbols per period. That is, only one state transition out of state S_i into state S_j. If this is the case then the sum of the forward variables can only equal 1 implying a perfect model. Mathematically, we can write

$$\alpha_{t+1}(j) = \begin{cases} 1 & if \exists a\, set\, \{\alpha_t(i), a_{ij}, b_j(o_{t+1})\}\, such that\, \alpha_t(i) a_{ij} b_j(o_{t+1}) \langle 1 \\ p & 0 \le p \le 1 \end{cases} \quad (5)$$

On the other hand, if the number of states exceeds that of the number of symbols per period there has to be at least one state containing more than one transition and thus element a_{ij} that is not equal to 1. Although this results in a forward variable less than 1, the forward-backward procedure sums over all possible paths so that the sum of the terminal forward variables does not equal 1. The extra states can thus be viewed as redundant since they do not add any extra information to be model (but do not take any away either).

The perfect model, however, has one serious drawback when it comes to observation sequence recognition. Since the perfect model has a training probability of 1, it is only capable of reproducing one sequence. If we use the perfect model for recognition of test sequences that differ from the training sequences by even one symbol, the perfect model will collapse. The perfect model, in other words, lacks robustness.

3. RECOGNITION PERFORMANCE WITH ERRORS

Although passive ESM receiver is much less likely to receive error-corrupted radar. The active radar due to smaller free-space losses, an observation sequence with still invariably contains some errors. Therefore, it is necessary to have a model for given radar that is able to cope with certain error-level. In other words, it is required that our HMM be robust enough to counteract the effect of errors that occur from both spurious (extraneous) and dropped pulses. Dropped pulses are generally more common than spurious pulses and can occur for several reasons including: (1) transmitter malfunction, (2) reflecting objects that blind the receiver, (3) large distances between transmitter and receiver resulting in low SNR. Spurious pulses are rare and usually the result of internal processing at the receiver. The most common reason for spurious pulses is improper deinterleaving.

In speech recognition errors are usually interpreted as differences between the ways in which various speakers utter a given word [6]. Even the same person never be able to identically repeat a word twice, which is why training usually requires the person to say a given word multiple times. This use of multiple training sequences allows the HMM a certain robustness during recognition. It was initially thought that using multiple training sequences, which include a so-called perfect sequence (the original sequences) as-well as error corrupted versions, would also be an adequate way for improving recognition for RSI. However, it was observed that the multiple sequences might have adverse effect of removing information from a model that

was just trained with perfect sequences. That is to say, a perfect model contains all the correct PRI information about a given training sequence so it would be desirable to some how preserve some of this information while at the same time improving the robustness of the HMM. One way f doing this would be to train the model with the perfect training sequence in order to obtain a perfect model and then retrain only the ***B*** matrix (holding ***A*** and π constant) with an error corrupted sequence. This would preserve the information in the state transition matrix while perturbing the values the values in ***B*** a slight amount from the perfect model there by giving additional flexibility to HMM to accommodate errors in the observation sequences. Although one could do the reverse and retain ***A*** and π while holding ***B*** constant, ***B*** seems to be a much more powerful component of the model[6]

Figure 6 shows the results of retraining ***B*** in terms of log training probability for the sequence [0010100101…..] Initially, the perfect model was computed (using five states) and then the error level in the training sequence was varied from 0% to 100% to give the corresponding retrained probability $P(O|\lambda)$.

Notice that perfect training probability occurs at 0 and 100% error levels while the worst probability occurs at 50%. In other words, the model works best where either none or all of the symbols are in error where as a 50% error rate means that there is no information about the sequence contained with the training sequence.

Next, a model was retrained according to fixed error level and then used to compute a recognition probability of an error-corrupted version of the original training sequence with an error level ranging from 0% to100%. In The figures 7, 8 , 9 and 10 shows the recognition performance of these models as compared to the perfect model trained on error-free sequence [00101] repeated eight times. Note that the perfect model was faced to have a ***B*** matrix consisting not of ones and zeros, but of ε and 1-ε with ε equal to 0.0001. This was done so that the simulation could compute recognition probability without producing “NaN” a Matlab expressions for operations such as $\frac{0}{0}$ or $\frac{\infty}{\infty}$. Additionally if any of the elements of the retrained models ***B*** matrix contained either a ‘0’ or ‘1’ then these elements were also changed to ε and 1-ε respectively.

It is clear from the previous four figures that adjusting ***B*** to accommodate an error-corrupted test sequence certainly, on the whole, provides a better recognition probability than the perfect model as would be expected. The model retrained at 50% error gives relatively constant performance across the range of the observation errors while the other models are somewhat mirror images of each other. That is to say that the model trained on 35% error level has better recognition for observation sequences containing less than 50% errors while that model trained on 85% has better recognition for test sequences having more than 50% errors.
Using the recognition with multiple competing models using the retraining

method described in the previous sections, the recognition performance of a HMM is tested with competing models. The fundamentals equation at hand is how corrupted can an observation sequence be before it is mistakenly classified by wrong model. In order to investigate this issue three different artificial radar signals are created and their respective perfect HMMs computed using number of states as specified in section2.
The three radar sequences used are the following (each observation is 70 symbols in length):, (1)Radar 1: [0101…], (2)Radar 2: [0010100101…], (3)Radar 3: [0001001 0001001…].

The error level at which each HMM is retrained is varied from small (around 5%) to large (up to 60%). The observation sequence used is generated from radar 2 and it has an error level that ranges from 0 to 100%. In other words, this experiment is attempting to uncover how well a model retrained at known error level would perform where observation sequence is error corrupted.

Figure 11 shows the log recognition probabilities for each of the three HMMs. The error level at which each HMM is retrained is shown on the top of the figure while the error level of the observation sequence is shown on the x-axis. The retraining error level for radar 1 is abbreviated as EL-1, radar 2 as EL-2 and radar 3 as EL-3 It can be seen that correct recognition will occur in this case up until the error level of the observation sequence reaches approximately 18%. After this the highest log recognition scores vary between models 1 and 3, which tend to have a relatively flat recognition probability curve.

Figure 12 increases the retraining error level for each of the models to around 25%. The reason they are not all the same is that the number of errors in each retraining sequence is determined randomly. That is, a uniform pseudo-random variable is generated for every symbol in the sequence by the computer and it is below a given threshold the symbol is inverted (a '0' changed to a '1' and vice versa). In this case threshold is 0.25. In this simulation, correct recognition tends to occur up until approximately 32% error level in the observation sequence- slightly than in Figure 11, which is expected because the retraining error level is high than previously. Notice also that the recognition probabilities for the two competing models have risen substantially.

Finally, Figure 13 shows the recognition performance for each HMM after having been retrained on very corrupted sequences. Notice how model 2 only begins to have the highest recognition probability of the three when observation sequence also contains many errors. This is similar to Figure 9 where the model has been retrained at 85% error and whose recognition probabilities increased as the observation sequence error level increased.

Of course, it is rare that observation sequences will be so corrupted that 80% of the sequences is in error. In radar applications common assumption for dropped and spurious pulse rate are 10% and 50% respectively. For example, if a transmitter signal contain 40 pulses ('1's) and 100_8empty cycles ('0's) then one would expect

that, an average, $40 \times 0.1 = 4$ pulses will be dropped (changed to '0's) and $40 \times 0.05 = 2$ empty cycles will be received pulses (changed to '1's). in this case there will be a total o 6 errors or $\left(\frac{6}{140}\right) \times 100\% \cong 4.3\%$ error.

4. UNIFORM INITIAL STATE DISTRIBUTION VECTOR AND LOSS OF SYNCHRONIZATION

Up until this point the initial state vector π, has been of the form

$$\begin{aligned} \pi_i &= 1 \qquad\qquad i = k \\ &= 0 \qquad\qquad 1 \le i \le N \qquad i \ne k \end{aligned} \tag{6}$$

Where the index k is arrived at using the Baum Welsh algorithm. The reason for this is that one of the elements in π must exactly equal 1 in order to have perfect model. It has already been seen that the perfect model is in flexible in classifying error-corrupted sequences and that its ***B*** matrix entries must be adjusted by $1-\varepsilon$ and ε to improve its recognition performance. Another way to further increase its robustness is to change the initial state distribution so that the model can begin in any of its N states with equal probability. This is essential for observation sequences that are not synchronized; that is, the final received symbol in the periodic sequences. For example, consider a model trained on the sequence [0010100101…] that it has the following HMM parameters and state diagram in Figure 14

$$\begin{aligned} A &= \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \\ B &= \begin{bmatrix} 1-\varepsilon & 1-\varepsilon & \varepsilon & 1-\varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1-\varepsilon & \varepsilon & 1-\varepsilon \end{bmatrix} \\ \pi &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{aligned} \tag{7}$$

Now we want to see the effect on the recognition probability for sequentially shortening the observation sequence from the left symbol by symbol. In other words, if the first observation sequence is 0010100101 (2 periods) then the next sequence would be 010100101 followed by 10100101, 0100101 and so on. This is, in effect, simulating receiving a signal that is in mid-transmission. Figure 15 shows the log recognition probability for each length of the observation sequence.

The oscillatory nature of Figure 15 can be explained as follows. The log recognition probability of the 10 symbol test sequence is very nearly equal to 0 (more precesizely) $\log(1-\varepsilon)^{10}$ because this^{9}sequence matches up exactly to the

training sequence the HMM is designed to track. The 9 symbol sequence [010100101], however, faces the HMM to use some of the ε values in ***B*** because this sequence does not line up perfectly with the training sequence. Table 1 illustrates which elements in ***B*** are hit for the 5 to 10-symbol observation sequences. The element in ***B*** that is used depends on whether the symbol at time t for the standard sequence is the same as the original training sequence (the top sequence). If it is then HMM will use $1-\varepsilon$=0.9999 other wise it will use ε=0.0001. the total probability of tracking the sequence, $P(O|\lambda)$ is the product of all ***B*** element used. This is why the recognition probability of the 10-symbol is $(1-\varepsilon)^{10}=0.9999$. The log probability of the 9-symbol sequence can be calculated as $2\log(1-\varepsilon)+7\log\varepsilon=-28$ and so on. If the model is not constrained to begin in state 1, but is instead, allowed to begin tracking in any state with equal probability $\left(\pi_i=\frac{1}{N}\ ;\ i=1,2,.....N\right)$ then we get Figure 16.

These sequences with 3 or more symbols have recognition probability equal to

$$P(O|\lambda)=\frac{1}{N}(1-\varepsilon)^T \tag{8}$$

Which in logarithm form, is

$$\log\ P(O|\lambda)=T\log(1-\varepsilon)-\log N \tag{9}$$

Where T is the length of the sequence. In this case each of these probabilities is roughly equal to -0.6990. the remaining two sequences can follow two different state sequences so their respective log recognition probabilities are larger by log 2 or 0.301 since the forward backward procedure sums over all possible state sequences. That is

$$\log P(O|\lambda)=\log\left(\frac{1}{N}(1-\varepsilon)^T\times 2\right)=T\log(1-\varepsilon)-\log\frac{N}{2} \tag{10}$$

Regardless of whether a given observation sequence is able to follow more than one state sequences it is clear that a uniform initial state vector will provide overall better recognition performance then if it is constrained to beginning I only one state. If we assume that an unsynchronized observation sequence of length T can only follow one state sequence then it is possible to define the following improvement factor for using a uniform π .

$$I = \frac{P\left(O\middle|\lambda_{uniform}\right)}{P\left(O\middle|\lambda_{nouniform}\right)} = \frac{\left(1-\varepsilon\right)^{T}}{N\left(1-\varepsilon\right)^{T-x}\varepsilon^{x}} \tag{10}$$

After some simplification and taking the logarithm becomes

$$\log I = x\log\left(\frac{1-\varepsilon}{\varepsilon}\right) - \log N \tag{11}$$

Where x is the number of symbols in the sequence that hit the ε elements in the ***B*** matrix during the recognition. As an example, using 9-symbol sequence 010100101 that has $x = 7$ we get a log improvement equal to 27.3007.Because a uniform π is so much flexible for sequence recognition.

5. CONCLUSIONS

The performance of HMMs with deterministic sequences has been investigated. It was found that a perfect model can readily found for any HMM trained on a deterministic sequences by choosing the number of states to be at least as long as the period of the sequence. Significant robustness can be added if the elements in the symbol distribution matrix ***B***, are changed from '1's to '0's to $1-\varepsilon$ and ε respectively where ε is a small value. Additionally, the actual recognition probability of error-corrupted sequence can be increased if the HMM is retrained on a sequence which also contain a known number of errors. The sequences used in this paper, however showed that the recognition performance of a HMM with competing models may degrade after retraining if the error rate of the observation sequence significantly exceeds that of the retraining sequence. This is due to the fact that the recognition probabilities of the competing models increase substantially more than those of the correct model.

Finally, it was shown that one can further increases a model's robustness by using a uniform initial state distribution vector. The log improvement factor was determined***.*** If x=0 then the recognition probability decreases proportionally to the logarithm of the number of states, ***N***, however, it is very rare that the observation sequence will be completely error-free and synchronized.

6. REFERENCES

[1] Rabiner, L.R., "A Tutorial on Hidden Markov Models and Applications in Speech Recognition," Proc. IEEE, Vol.77, NO.2 February. pp 257-285, 1989.

[2] Roe,D.B., Wilpom,J.G."whither speech recognition: the next 25 years", IEEE comm. Magazine pp. 54-62 November, 1993

[3] Baker, J.K., "the dragon system- an overview", IEEE trans. On Acustics, speech, and signal processing, vol. Assp-23 NO.1,pp 24-29, February, 1975.

[4] Picone, J.W., "signal modeling techniques in speech recognition", proc. IEEE, vol.81,No.9 p 1261 September, 1993

[5] Matsui, T., Nishitani, T., Firui, S., "Robust methods of updating model and a priori threshold on speaker verification",proc.21st IEEE Int. conference on Acustics, speech and signal processing, Atlanta, GA,7-10 May, 1996.

[6] Levinsons, S.E., rabiner, L.R.Sondhi,M.M., "An introduction to the application of probabilistic functions of a Markov process to automatic speech recognition", the Bell system technical journal, vol.62 No.4,pp.1041-1050 April, 1983.

[7] Mardia, H.K., " New techniques for deinterleaving repetitive sequences," IEE proc. F, comm., Radar& signal processing 136, (4), pp. 149-154,1989.

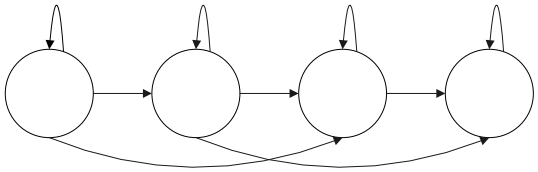

Figure 1. 4-state left-right HMM

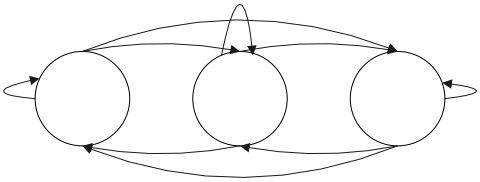

Figure 2. 3-state ergodic HMM

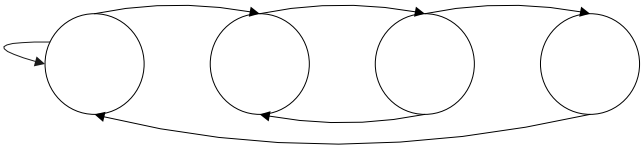

Figure 3. 4-state partially ergodic HMM

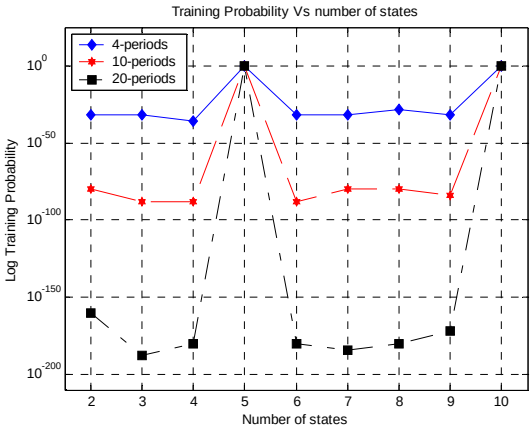

Figure 4. Training probability Vs number of states

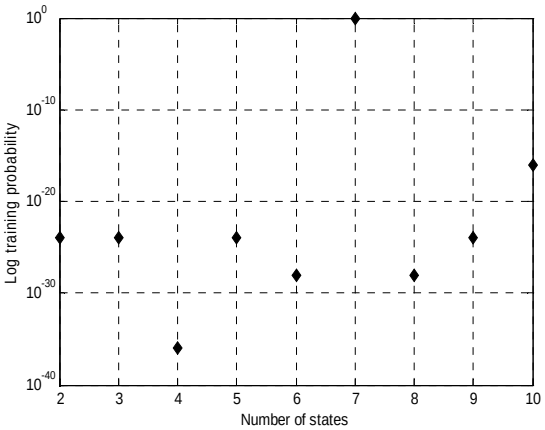

Figure 5. Training probability Vs number of states

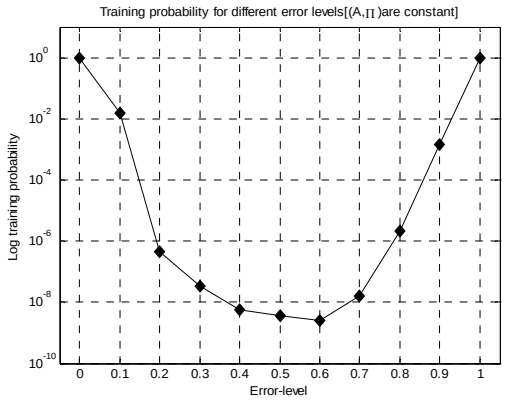

Figure 6. Training probability with different training sequence error levels

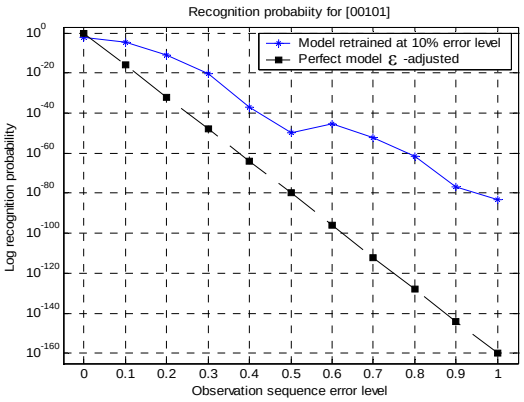

Figure 7. Recognition probability for HMM retrained at 10% error level

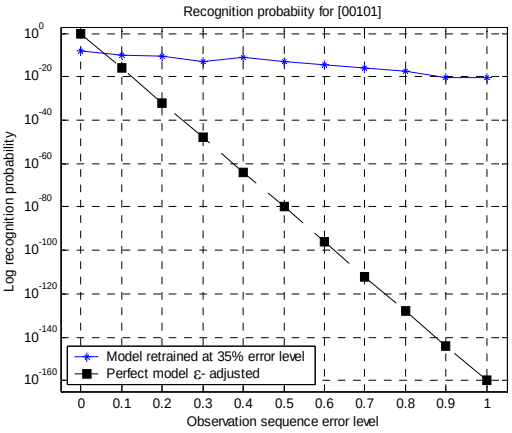

Figure 8. Recognition probability for HMM retrained at 35% error level

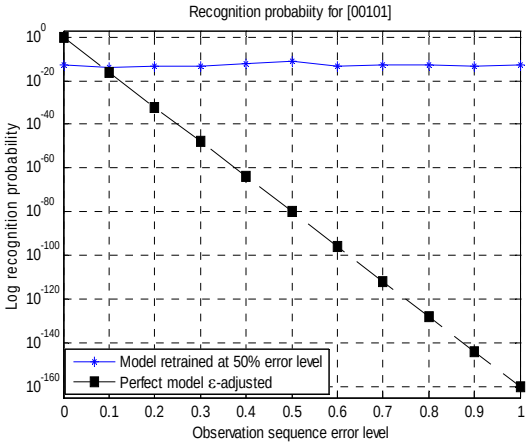

Figure 9. Recognition probability for HMM retrained at 50% error level

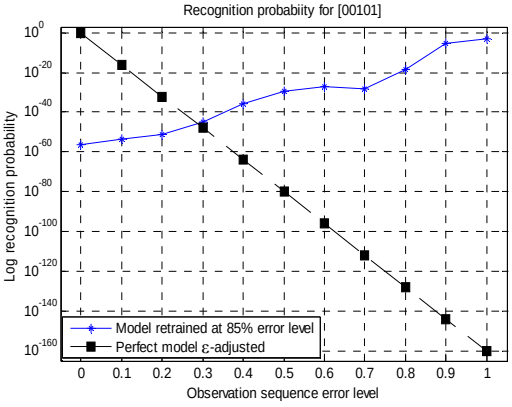

Figure 10. Recognition probability for HMM retrained at 85% error level

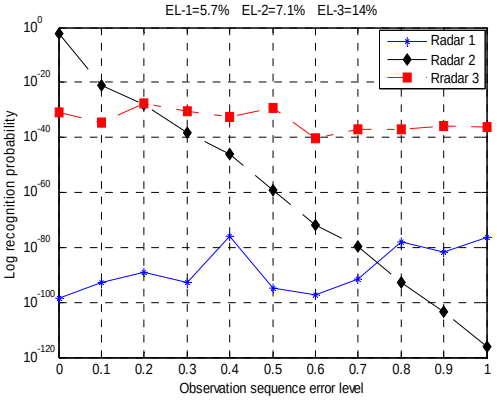

Figure 11. Recognition probabilities with multiple HMMs (low-error retraining/ radar2 sequence generator)

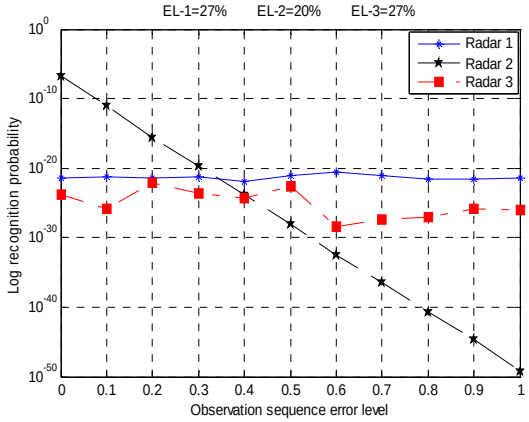

Figure 12. Recognition probabilities with multiple HMMs (medium error retraining/ radar 2 sequence generators)

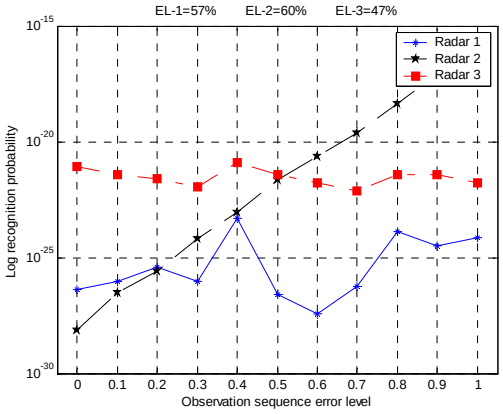

Figure 13: Recognition probabilities with multiple HMMs (higher error retraining / radar 2 sequence generator)

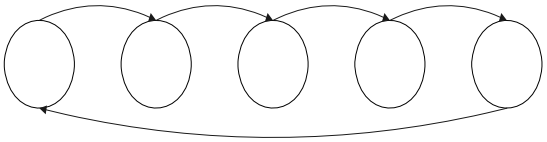

Figure 14. state perfect HMM

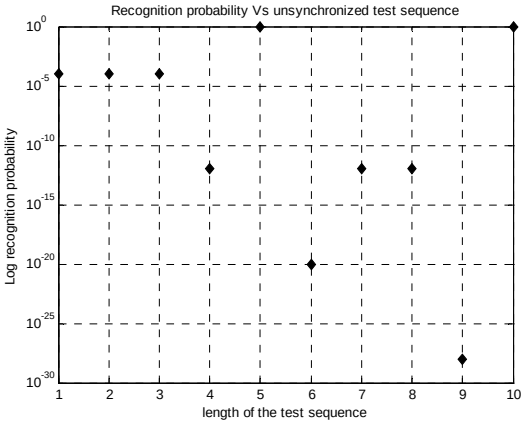

Figure 15. Effect of loss of synchronization on recognition probability

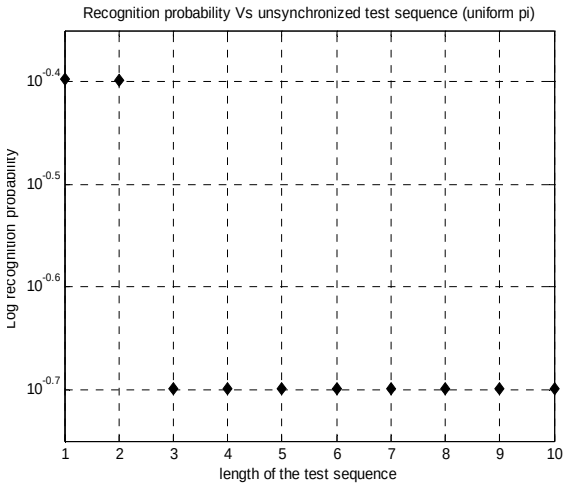

Figure 16. Improvement in recognition probability with

Table 1. Recognition probability calculation

Sequence	0	0	1	0	1	0	0	1	0	1
B element used	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$
Sequence	0	1	0	1	0	0	1	0	1	
B element used	$1-\varepsilon$	ε	ε	ε	ε	$1-\varepsilon$	ε	ε	ε	
Sequence	1	0	1	0	0	1	0	1		
B element used	ε	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	ε	ε	$1-\varepsilon$	$1-\varepsilon$		
Sequence	0	1	0	0	1	0	1			
B element used	$1-\varepsilon$	ε	ε	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	ε			
Sequence	1	0	0	1	0	1				
B element used	ε	$1-\varepsilon$	ε	ε	ε	ε				
Sequence	0	0	1	0	1					
B element used	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$	$1-\varepsilon$					