homework solutions for:

Homework #6: Hidden Markov Modeling

submitted to:

Dr. Joseph Picone
ECE 8993 Fundamentals of Speech Recognition

May 20, 1998

submitted by:

Jonathan Hamaker

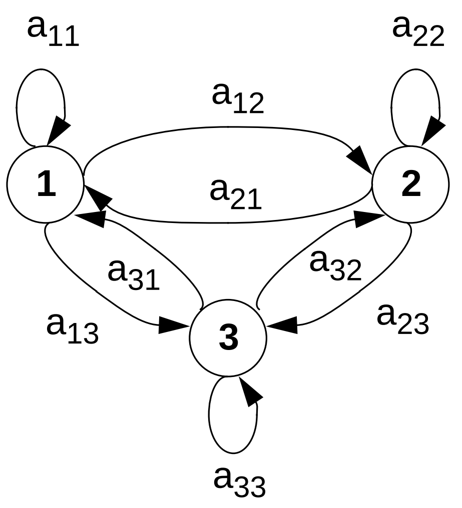
Institute for Signal and Information Processing
Department of Electrical and Computer Engineering
Mississippi State University
Box 9571, 216 Simrall, Hardy Rd.
Mississippi State, Mississippi 39762
Tel: 601-325-8335, Fax: 601-325-3149
Email: hamaker@isip.msstate.edu

Introduction

One of the most important innovations in speech recognition research in the past half-century is the integration of Hidden Markov Models (HMMs) as the core of the acoustic model. HMMs are a doubly stochastic model which have the distinct advantage of being able to “learn” the data. By “learn” we mean that, through an appropriate training regimen, the model determines the optimal parameter set to represent the training data — i.e. they maximize the probability of the data given the model, $P(O|M)$. It is the job of the model to represent the variance inherent in the data.

The term “hidden” comes from the fact that, for any sequence of observations produced by the model, it is hidden from the observer what sequence of states in the HMM generated that sequence. This is another advantageous property as it is now not the job of the system designer to determine what parts of the model are mapped to a particular state in the model. A picture of a fully ergodic (fully connected) HMM is shown in Figure 1. Note that the key characteristics of the HMM are the output probability distribution, $\boldsymbol{B}$, the transition probabilities, $\boldsymbol{A}$, and the start state probabilities, π.

In this assignment we use HMMs to model a simple coin toss game. In this game a coin or group of coins are repeatedly tossed to produce the training data. From this data we build a model of the coin which produced the training data. We test this data on new sequences of data (possibly from a different set of coins: akin to speaker-independent speech recognition) to see how well our trained model can predict the test data. A more detailed treatment of Hidden Markov Model theory is given in [1] and [2].



P(H):	P_1	P_2	P_3
P(T):	$1-P_1$	$1-P_2$	$1-P_3$

$$S = 3$$

$$\pi(1) = \begin{Bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{Bmatrix}$$

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} P_1 & P_2 & P_3 \\ 1-P_1 & 1-P_2 & 1-P_3 \end{bmatrix}$$

Figure 1. A simple, fully ergodic Hidden Markov Model

Problem Statement

Using the discrete HMM tool, train the best model you can for the following data sequence:

```
HHTTTHHHHTTTTTHHHHHHTTTTTTTHHHHHHHHTTTTTTTTTHHHHHHHHTTTTTTTH
HHHHHTTTTTHHHHTTTHHTHHTTHHHHHHHHHHTHHHHHTHHHHHHHHHHTHHHTHHHH
HHHHHHHHHTHHHHHHHHHTTHHHHHHTHHHTHHHHHTHHHHHHHHTHHHHHHTHTHHHH
HHHHHHHTTHHHHHHHHHHHHHHHTHHHHHHTTHHHHHHHHHTTHHHHHHHHHHHHHTTH
HHHHHHHHHHHTTHHHTHHHHHHHHHHHTHHHHHHTTHHHHHTHHHTHHHHHTHTHHHHH
THTHHHHHHHHHTHTHHHHHHTHTHHHHHTHTTHHHHHHHHHHHHHTHTHHHHHHHHHHH
HTHHHHHHHHHHHHHHHTHHHHHHHHHHHHHHHTHHHHHHHHHHTTHHHHHHHTHHHHHH
HHHHHTHHHHTHHHHHHHHHHTHHHHHHHHHHHHTTHHHHTTHHHHHHHHHHHHHHTTHH
```

With this model, compute the probability of the following sequences:

```
HTHTHTHTHTHTHTHTHTHT
HHHHHHHHHHHHHHHHHHHH
TTTTTTTTTTTTTTTTTTTT
```

Methodology

Our first inclination is to do an exhaustive search of all possible HMM configurations using the tool — after all CPU cycles are cheap. However, the more one deals with problems in speech research, the more one realizes how important it is to understand the data. Table 1 gives some rudimentary statistics of the training data.

The most important thing to see in this data is how skewed it is towards picking heads. Notice that an “H” occurs almost 4 times more often than a “T”. So, what does this imply? Looking at the training data, this should be clear. Even if we can train a model that recognizes the training data perfectly, how well do we think it will do on the test sample that is all “T”s? Not very well considering that the likelihood of encountering a “T” is low and the likelihood of encountering multiple consecutive “T”s is extremely low. On the other hand, we expect that our model will do a pretty good job on the utterance with only “H”s since the likelihood of encountering an “H” is high and the likelihood of encountering multiple consecutive “H”s is also high.

Thinking about this in terms of an actual coin game, if we were betting on this game and we had *a priori* knowledge of the coin being used we would have high confidence that this coin could produce a sequence of 20 heads. Thus, we would say that it is very likely that a sequence of 20 heads could come from this coin. We would also have very low confidence that the coin could produce a sequence of 20 consecutive tails. Thus, we would attribute a low likelihood to the coin producing 20 tails. What about the sequence of alternating heads and tails? Certainly this is more likely than have a sequence of only tails and it is certainly less likely than having a sequence of only heads. So we might attribute a moderate level of confidence in obtaining this sequence from the given coin.

Given this knowledge we can assume that our experiments will fail to produce a good model to explain the test data — let’s verify it! We are using a discrete HMM tool developed at the Institute for Signal and Information Processing to train and test our models [3]. This tool uses fully ergodic

Occurrences of “H”	380
Occurrences of “T”	100
Occurrences of a “H” to “T” transition	54
Occurrences of a “H” to “H” transition	318
Occurrences of a “T” to “T” transition	45
Occurrences of a “T” to “H” transition	55

Table 1: Statistical counts of the training data. Notice how much more likely it is that the given coin will produce a “H” than a “T”

models with the initial values randomly assigned. With this tool, we have the option of training and testing using either the Viterbi or Baum-Welch algorithms. We can also vary the number of states in the model, the number of training passes, and the number of output symbols. Our goal is to find model which best represents the training data and to use that model to score the testing data. Since we believe that this score will be bad, we’ll also train the worst model we can and use that to score the testing data. We will also use just the Viterbi training algorithm as it has more stable performance on our data.

Results

As a first step in our experiment we found the model which best represented the training data and the one which worst represented the training data by doing an exhaustive sweep of number of states (varying from 1 to 20) and number of output symbols (varying from 2 to 8). For each combination we are using 10 training passes to insure convergence. From this exhaustive search, we found the model which best represented the data, the one which did the worst job of representing the data and the median-scoring model. Table 2 shows the results of these three samples. It is reassuring to see that the best performing system only used two output symbols.

Next we tested all three of these models using the testing data and both the Viterbi and Baum-Welch methods. The results of this experiment are shown in Table 3. The first trend to notice from these results is that the Baum-Welch testing and Viterbi testing produce vastly different results on the same data. With Viterbi decoding, we see that our intuition was right in most cases that the heads-only test vector would score best, the tails-only would score worst and the toggled vector would score somewhere between those two. What is somewhat puzzling is the rankings of the best, worst, and median models when using Baum-Welch testing. The Baum-Welch testing shows that the median model does a better job of balancing all possibilities and that even the worst model does a better job than the best model.

References

[1] J. Picone, “Fundamentals of Speech Recognition: A Short Course,” Institute for Signal and Information Processing, Mississippi State University, 1996.

[2] F. Jelinek, *Statistical Methods for Speech Recognition*, The MIT Press, Cambridge, Massachusetts, 1997.

[3] J. Picone, “A Discrete Hidden markov Model (HMM) Demonstration,” Institute for Signal and Information Processing, *http://www.isip.msstate.edu/resources/technology/software/1996/discrete_hmm.*

	Best	Median	Worst
# of states	3	5	15
# of output symbols	2	3	7
P(O/M)	-97.958	-106.782	-147.797

Table 2: Parameters for best, median, and worst performing models on training data

	Best	Median	Worst
Overall score with Viterbi testing	-20.190	-23.494	-29.624
Score on heads-only test vector (Viterbi testing)	-1.344	-2.003	-3.277
Score on toggled test vector (Viterbi testing)	-10.874	-7.831	-6.423
Score on tails-only test vector (Viterbi testing)	-7.971	-13.629	-19.925
Overall score with Baum-Welch testing	-28.774	-20.011	-25.154
Score on heads-only test vector (Baum-Welch testing)	-4.711	-1.855	-2.114
Score on toggled test vector (Baum-Welch testing)	-13.676	-7.227	-5.989
Score on tails-only test vector (Baum-Welch testing)	-10.387	-10.930	-17.051

Table 3: Results from testing models which had the best training score, worst training score, and median training score.