Hidden Markov Models

Connor Baugh

3/13/2022

1 Introduction

Hidden Markov Models (HMMs) are a type of graphical model used to predict unobserved discrete "hidden" states given observations of some Markov process. They are a natural extension of Markov chains, a model which can predict states given only the previous state observed, but which are incapable of modelling "hidden" states with unknown probabilities given the same observations. HMMs are able to achieve this by deriving the probabilistic features of a process through the use of probability distributions that explain the relationships between observable variables and unknown hidden states [Ver21]. HMMs are considered a special case of Dynamic Bayesian Networks (DBNs) because of the assumption that there is only one discrete hidden state as opposed to the arbitrarily many state variables that can be modelled using DBNs [299]. Because they only predict a single hidden state for each time step of a Markov process, HMMs are technically the simplest possible form of a DBN.

Since their introduction in the late 1960's by Dr. Leonard Baum, Hidden Markov Models have been applied in a wide array of industries, most notably in the fields of speech recognition and bioinformatics [BP66]. The field of speech recognition in particular saw some of the earliest adoptions of HMMs with the design of a linguistic statistical decoder and the introduction of the DRAGON System for continuous speech recognition in 1975, the former of which closely mirrors the framework of many HMMs used for speech recognition today [Bak75, JBM75]. Because of their simpleness and effectiveness within the domain over the last 5 decades, nearly all current large scale continuous speech recognition systems are now HMM-based [GY07].

In this paper we aim to highlight the effectiveness of Hidden Markov Models in the domain of continuous speech recognition systems. We utilize the AudioMNIST dataset, which comprises a set of free spoken digits between 0 and 9, to train an HMM for each digit class to predict an appropriate sequence of phonemes. Additionally, we evaluate the classification accuracy of each HMM on a hold out set of the respective digits. In Section 2, we present the methodology to construct an HMM. We then preprocess the data, run the experiment and analyze the results in Section 3.

2 Methodology

2.1 Defining the HMM

Let the observed sequence be denoted as $\mathbf{x} = x_1, x_2, \cdots, x_L$ and the state sequence be denoted as $\mathbf{y} = y_1, y_2, \cdots, y_L$, where y_n corresponds with the hidden state of the observation x_n. Each state y_n is assigned a value from the state set $\mathbf{S} = \{1, 2, \cdots, N\}$, where N equals the total unique states in the model [Yoo09].We assume that the hidden state sequence satisfies the Markov property such that state y_n depends only on the previous state, y_{n-1} [Deg14]. With this assumption, we can now represent the probability of transitioning from state i to state j as

$$P\{y_{n+1} = j | y_n = i, y_{n-1} = in - 1, \cdots, y_1 = i_1\} = P\{y_{n+1} = j | y_n = i\} = t(i, j) \tag{1}$$

for all hidden states $i, j \in \mathbf{S}$ and $n \geq 1$. This is the *transition probability.* Additionally, we represent the *initial state probability* as $\pi(i) = P\{y_1 = i\}$ for all $i \in \mathbf{S}$ [**?**]. Because the probability that $x_n = x$ only depends on y_n, we can now represent it as

$$P\{x_n = x|y_n = i, y_{n-1}, x_{n-1}, \cdots\} = P\{x_n = x|y_n = i\} = e(x|i) \tag{2}$$

for all state $i \in \mathbf{S}$ and all $n \geq 1$. This is the *emission probability.* Given the three probabilities $t(i, j)$, $\pi(i)$, and $e(x|i)$, we can now entirely define an HMM. For organization purposes, we can represent this set of probabilities as Θ. With that, the probability that the HMM generates the observation sequence $\mathbf{x}$ given the hidden state sequence $\mathbf{y}$ can now be calculated as

$$P\{x, y|\Theta\} = P\{x|y, \Theta\}P\{y|\Theta\}, \tag{3}$$

where

$$P\{x|y, \Theta\} = e(x_1|y_1)e(x_2|y_2)\cdots e(x_L|y_L), \tag{4}$$

and

$$P\{y|\Theta\} = \pi(y_1)t(y_1, y_2)t(y_2, y_3)\cdots t(y_{L-1}, y_L). \tag{5}$$

[Yoo09]

2.2 Forward Algorithm

Suppose we are given another observation sequence $\mathbf{x}$. Given this entirely new sequence, we now have no way to calculate $P\{x|\Theta\}$. This is commonly known as the *scoring problem.* The *forward algorithm* is a computationally efficient solution to this problem written as

$$P\{x|\Theta\} = \sum_k \alpha(L, k), \tag{6}$$

where the *forward variable* α is defined as

$$\alpha(n, i) = P\{x_1 \cdots x_n, y_n = i|\Theta\}, \tag{7}$$

and can be recursively computed with

$$\alpha(n, i) = \sum_k [\alpha(n-1, k)t(k, i)e(x_n|i)]. \tag{8}$$

[Yoo09]

2.3 Viterbi Algorithm

An additional problem that needs to be solved is the *optimal alignment* problem. That is, we want to find the hidden state sequence, or path, that best explains the given observation sequence where the optimal path $\mathbf{y^*}$ is defined as

$$y^* = \arg\max_y P\{y|x, \Theta\} \tag{9}$$

However, calculating the optimal hidden state sequence $\mathbf{y^*}$ by searching through all N^L possible paths is computationally impractical. To get around this, we can use another computationally efficient solution known as the *Viterbi algorithm.* The Viterbi algorithm allows us to to calculate the maximum observation probability

$$P^* = \max_y P\{x, y|\Theta\} = \max_k \gamma(L, k), \tag{10}$$

by defining the variable

$$\gamma(n, i) = \max_{y_1,\cdots,y_{n-1}} P\{x_1 \cdots x_n, y_1 \cdots y_{n-1} y_n = i|\Theta\}, \tag{11}$$

and recursively computing it with

$$\gamma(n, i) = \max_k [\gamma(n-1, k)t(k, i)e(x_n|i)]. \tag{12}$$

[Yoo09]

2.4 Backward Algorithm

Although the Viterbi algorithm is able to find the optimal states that maximize the probability of the observations over the whole sequence, there may be a case in which it is better to calculate the optimal state $\hat{y_n}$ for each individual observation such that

$$\hat{y_n} = \arg\max_i P\{y_n = i|x, \Theta\}, \tag{13}$$

where

$$P\{y_n = i|x, \Theta\} = \frac{P\{x_1 \cdots x_n, y_n = i|\Theta\} P\{x_{n+1} \cdots x_L|y_n = i, \Theta\}}{P\{x|\Theta\}} = \frac{\alpha(n, i)\beta(n, i)}{\sum_k \alpha(n, k)\beta(n, k)}, \tag{14}$$

and the *backward variable* $\beta(n, i)$ is denoted as

$$\beta(n, i) = P\{x_{n+1} \cdots x_L|y_n = i, \Theta\}. \tag{15}$$

From there, we can recursively compute the $\beta(n, i)$ with the *backward algorithm* such that

$$\beta(n, i) = \sum [t(i, k)e(x_{n+1}|k)\beta(n+1, k)], \tag{16}$$

[Yoo09]

3 Illustrative Example with R

3.1 Preprocessing Audio

The AudioMNIST dataset is composed of 3,000 .wav files with digits between 0 and 9 spoken aloud by 6 people of varying age and gender. As such, we must convert the dataset into digital representations of each audio file. An example of the digitized representation after this conversion is shown in Figure 1.

In order to get the data into a format appropriate for the HMMs, it must be further processed into *Mel Frequency Cepstral Coefficients* (MFCCs). MFCCs are better suited for HMMs because each MFCC roughly represents each phoneme we are trying to model [Nai18]. The same digit 3 shown previously is displayed in Figure 2 in MFCC form.

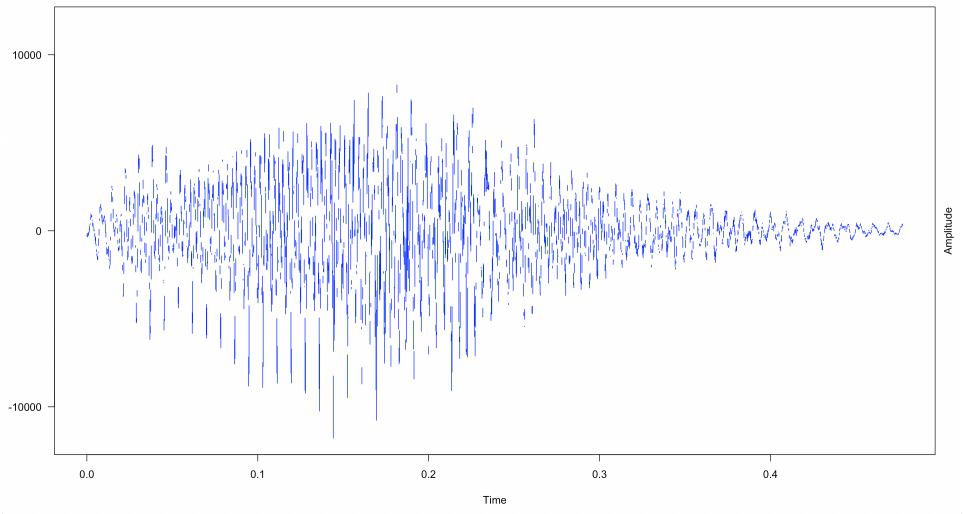

Figure 1: Waveplot of the digit 3

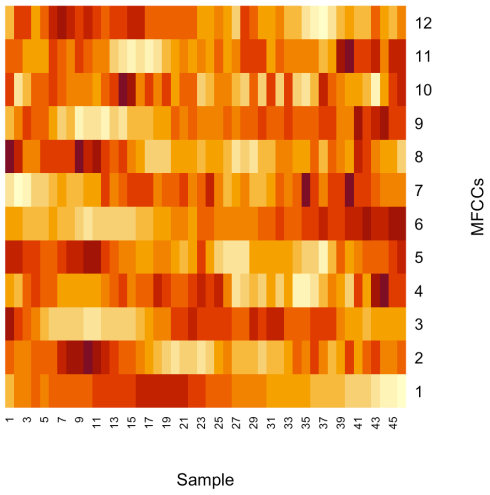

Figure 2: MFCC representation of digit 3

3.2 Speech Recognition with HMMs

After converting all of the audio samples into MFCCs, we are finally ready to train the HMMs for each digit. We have separated the samples into the appropriate training and validation sets for each of the respective digits and have fit 10 HMMs for each of the digit training sets with n states set depending on the number of phonemes present in the respective spoken digit. Each digit and it's associated phonemes can be found in Table 1 [Hui19].

Now that the models are fit, we can predict the states of the validation sets for each observation to get an idea of how well the models generalize to unseen samples. Figure 3 shows the predicted states of the same spoken digit from the digit 3 validation set against the waveplot of the data.

Digit	Phonemes
0	z iy r ow
1	w ah n
2	t uw
3	th r iy
4	f ao r
5	f ay v
6	s ih k s
7	s eh v ah n
8	ey t
9	n ay n

Table 1: Phonemes of each digit

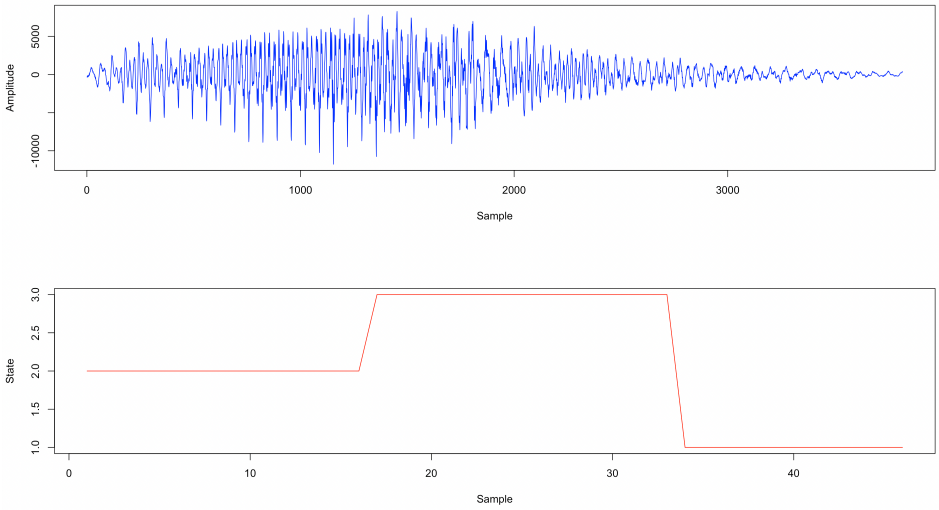

Figure 3: Predicted states vs Waveplot of digit 3 from validation set

3.3 Digit Classification and Results

To perform digit classification, we first calculate the maximum likelihood of each digit in the validation set fitting the distribution of each HMM. Then, we take whichever HMM receives the highest likelihood score and count that as the digit's assignment. Lastly, we use each digit's classification to record the accuracy of each HMM, defining accuracy as the number of digits correctly assigned to their respective class. The results of the digit classification are displayed in Table 2 below.

Digit	Val Acc
0	0.83
1	0.89
2	0.91
3	0.77
4	0.82
5	0.73
6	0.84
7	0.94
8	0.86
9	0.85

Table 2: Accuracy of each Digit HMM

4 Conclusion

In this paper we have provided an introduction to Hidden Markov Models and their processes and have highlighted their effectiveness in the domain of continuous speech recognition systems over the last 5 decades. We utilized the AudioMNIST dataset, comprising a set of 3000 samples of free spoken digits between 0 and 9, to train an HMM for each digit class to predict an appropriate sequence of phonemes. Additionally, we constructed a mechanism utilizing the maximum likelihood of each sample fitting the distribution of an HMM to perform digit classification and evaluated the classification accuracy of each HMM on a hold out set of digits. We found that, while not perfect, the Digit HMMs each did an acceptable job of classification on the unseen set of samples, with an average accuracy across all HMMs of approximately 85%.

References

[299] Dynamic bayesian networks (dbns), 1999.

[Bak75] J. Baker. The dragon system–an overview. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):24–29, 1975.

[BP66] Leonard E. Baum and Ted Petrie. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics*, 37(6):1554 – 1563, 1966.

[Deg14] Alperen Degirmenci. Introduction to hidden markov models, 2014.

[GY07] M. Gales and S. Young. The application of hidden markov models in speech recognition. *Foundations and Trends in Signal Processing*, 1(3):195–304, 2007.

[Hui19] Jonathan Hui. Speech recognition — kaldi, 2019.

[JBM75] F. Jelinek, L. Bahl, and R. Mercer. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, 21(3):250–256, 1975.

[Nai18] Pratheeksha Nair. The dummy's guide to mfcc, 2018.

[Ver21] Yugesh Verma. A guide to hidden markov model and its applications in nlp, 2021.

[Yoo09] Byung-Jun Yoon. Hidden markov models and their applications in biological sequence analysis. *Current Genomics*, 10(6):402–415, 2009.