Speech Recognition

Today, we will cover another application of probability: speech recognition. How does Siri understand what we are saying? We speak into the iPhone microphone, which samples the analog sound waveform. Now, the iPhone needs to figure out what we told it. The goal is to transform the analog waveform into the command "driving directions to Stanford" for example. There is a lot of randomness involved in this problem: the accent of the user, the different characteristics of his voice or the ambient noise etc.

We want to design a speech recognition algorithm. How do we approach this problem? We need to understand the structure of the spoken language. Each word is a natural unit into which we can decompose the sentence. We can also decompose the words into smaller units. From a phonetic point of view, a word is decomposed into phonemes, that can contain vowels and consonants. For example 'dr' in the word driving is a phoneme. There are around 30 or 40 phonemes in English. Therefore the problem becomes figuring out the sequence of phonemes in the sentence. The sequence of phonemes forms the set of random variables that we want to estimate.

Let \( Y_i \) be the \( i \)th phoneme in the sentence that we want to recognize. We have \( Y_1, \cdots, Y_n \) random variables representing the sentence we want to recognize. We need to relate the signal that we picked up on the microphone to this sequence of variables. Each phoneme roughly corresponds to 10ms. We chop the analog signal into 10ms intervals. Then, we take the signal in each 10ms interval and extract some key features from it. The relevant information contained in speech is most apparent in the frequency domain. So usually, a short window Fourier analysis is done on the sampled signal from each 10 ms time interval, and the corresponding Fourier coefficients are extracted. These coefficients serve as features in the frequency domain, or spectral information to describe the phoneme in that 10ms interval.

Typically there are multiple features for the signal in each 10ms interval, corresponding to multiple Fourier coefficients for example. But let us simplify the story by assuming there is only a single feature. Let \( X_i \) be the value of the feature in the \( i \)th interval. The system diagram can be represented as:

![System Diagram for Speech Recognition](image)

For simplicity of discussion, we will further assume that our language only has two possible phonemes, say \( a \) and \( b \). (It is a pretty boring language, but suffices to convey any information we want.) So each of the \( Y_i \)'s can be either \( a \) or \( b \).
We need to specify two things to be able to derive our speech recognition algorithm:

1. A prior probability on the input
2. A statistical relation between the phonemes and the features,

Let us start with (2), by describing the statistical relation between the phonemes and the features. We assume that:

\[
X_i = \begin{cases} 
\mu_a + Z_i & \text{if } Y_i = a \\
\mu_b + Z_i & \text{if } Y_i = b 
\end{cases}
\]

where \(Z_i \sim \mathcal{N}(0, \sigma^2)\). This model can be interpreted as the feature having a mean value depending on which phoneme \(a\) or \(b\) is uttered, but with a random component due to, say, the environmental noise. Moreover, let us assume that the noises \(Z_i\)'s are mutually independent. Note that this model of the statistical relation between \(Y_i\) and \(X_i\) is the same as the one we used for the spam detection example in the last lecture. Just like in that example, we can estimate \(\mu_a\) and \(\mu_b\) if we have training data \((x_i, y_i)\)'s. This can be obtained, for example, by having recording the audio signals of known speech.

Now, let's go back to (1): how do we model the input sequence \(Y_1 \cdots Y_n\)? One natural way of doing this is assuming that these random variables are independent. We tried this trick before many times, like with \(n\) independent coin flips for example. However, this is not a good idea in this application. Some phonemes are more likely to follow other phonemes. For example, the phoneme \(th\) is more likely to be followed by an "e" rather than a "s". So assuming the random variables to be independent seems like a very poor model.

On the other hand, to specify a fully general joint distribution for \(Y_1, Y_2, \ldots Y_n\) needs \(2^n\) number of parameters, even for our very simple language. How are we going to capture the dependency, while still having a small number of parameters in our model? Remember, we need to estimate these parameters from the data, so the fewer number of parameters, the easier it is to estimate them.

Consider \(P(Y_n = y_n|Y_1 = y_1 \cdots Y_{n-1} = y_{n-1})\). This is the probability conditioning the present upon the past. We are going to assume that the dependence of \(Y_n\) on the past is entirely through the random variable \(Y_{n-1}\) that immediately precedes it. So the simplification we make is to suppose

\[
P(Y_n = y_n|Y_1 = y_1, \cdots, Y_{n-1} = y_{n-1}) = P(Y_n = y_n|Y_{n-1} = y_{n-1}) \quad \text{for all } y_1, \cdots, y_n.
\]

For \(n = 3\) for example, it reduces to

\[
P(Y_3 = y_3|Y_1 = y_1, Y_2 = y_2) = P(Y_3 = y_3|Y_2 = y_2) \quad \text{for all } y_1, y_2, y_3.
\]

This is equivalent to saying that \(Y_3\) is independent of \(Y_1\) conditional on \(Y_2\). This is called the Markov property, and the corresponding figure that illustrates the relation between the variables is

\[Y_1 - Y_2 - Y_3\]

This is called a graphical model, with the random variables as nodes of the graph. The interpretation of the graph is that when one disconnects the graph into two subgraphs \(G_1\) and \(G_2\) by removing a node \(Y_i\), then the random variables in \(G_1\) and \(G_2\) are independent conditional on \(Y_i\).

Going back to the speech recognition problem, recall we are assuming \(Y_n\) is independent of all the past given \(Y_{n-1}\). More generally, we will assume that \(Y_i\) is independent of all the past given \(Y_{i-1}\) for all \(i\). We can now write the joint probability distribution as:

\[
P(Y_1 = y_1, Y_2 = y_2 \cdots Y_n = y_n) = P(Y_n = y_n|Y_1 = a_1, \cdots, Y_{n-1} = y_{n-1})P(Y_1 = y_1, \cdots, Y_{n-1} = y_{n-1})
\]

\[
= P(Y_n = y_n|Y_{n-1} = y_{n-1})P(Y_{n-1} = y_{n-1}|Y_{n-2} = y_{n-2}) \cdots P(Y_2 = y_2|Y_1 = y_1)P(Y_1 = y_1)
\]
In other words, the relation between all our random variables is represented by the graphical model:

\[ Y_1 - Y_2 - Y_3 \cdots - Y_n, \]

where \( Y_{i+1} \) is independent of \( Y_{i-1}, \cdots, Y_1 \) given \( Y_i \). This is called a Markov chain. To specify a Markov chain, we need to specify:

1. **Transition probabilities**
   \[
P(v|u) := P(Y_i = v | Y_{i-1} = u) \quad \text{for all } u, v.
   \]

2. **Initial distribution**
   \[
   \pi(y) := P(Y_1 = y) \quad \text{for all } y.
   \]

Consider an example of a Markov chain with \( Y_i \in \{a, b\} \), shown in the figure below. Suppose it has the transition probabilities: \( P(b|a) = P(a|b) = 0.3 \), and \( P(a|a) = P(b|b) = 0.7 \). The value \( Y_i \) takes on is called the state of the system at time \( i \). If we want to predict how the Markov chain will go forward, we only need to know the state and not the past. The figure below is called the state transition diagram of the Markov chain.

Finally, we can put all the random variables of the problem into an overall graphical model for the speech recognition problem:

\[
x_1 x_2 \cdots x_{n-1} x_n
\]

\[
y_1 - y_2 - \cdots - y_{n-1} - y_n
\]

Note that if we remove the node \( Y_i \), the node \( X_i \) will be disconnected from the rest of the graph. This reflects the fact that the feature \( X_i \) depends on other random variables only through \( Y_i \).

How do we estimate the transition probabilities? We can look at texts and count the fraction of transitions from a to b versus from a to a, etc.

Now that we have a full model and we have estimated the parameters, the speech recognition problem is from the sequence of features \( X_i \)’s, we want to estimate the sequence of \( Y_i \)’s. We do not want the complexity of our algorithm to increase exponentially with the number of phonemes. Ideally the complexity should grow linearly. It turns out that while we motivated the Markov chain model from a statistical consideration, to minimize the number of parameters one needs to estimate, it turns out that the model has a vast computational advantage as well.

One last question about the model before we talk about the speech recognition algorithm: is the observation sequence \( X_1 \cdots X_n \) a Markov chain? No (why?). But the underlying sequence that we want to figure out is a Markov chain. That is the reason for calling this a Hidden Markov Model (HMM).