HMM Lecture Notes

Tuesday, October 29th

Dannie Durand and Rose Hoberman

1 Notation

1. N states ($S_1..S_N$)
2. M symbols in alphabet, Σ
3. parameters, λ:
   1. initial distribution of states $\pi(i)$
   2. transition probabilities $a_{ij} = P(q_t = S_i | q_{t-1} = S_j)$. Note that $\sum_{i=1}^{N} a_{ij} = 1, \ \forall j$
   3. emission probabilities $e_i(a)$ probability state i emits a
4. Sequence of symbols: $O = O_1, O_2, ..., O_T$
5. Sequence of states: $Q = q_1, q_2, ..., q_T$

2 Using HMM's for recogition

2.1 Questions to ask with an HMM

1. Given a sequence O, what is the true path? Otherwise stated, we wish to assign labels to an unlabeled sequence.
   Example: Identify the cytosolic, transmembrane, and extracellular regions in the sequence. In this case, we wish to assign the labels E, M, or C to the unlabeled data.
2. What is the probability that a given sequence O, was generated by the HMM?
   Example: Is the sequence a transmembrane protein?
3. What is the probability of being in state i when O_t is emitted?
   Example: Is a given residue localized to the membrane?
4. Use the HMM to simulate sequences.
   Example: Generate sequences with properties similar to real transmembrane sequences.

In the previous lecture, we started discussing algorithms for answering these questions.

1. We assume that most likely path, Q^*, is a good estimation of the true path, where Q^* is defined by

$$\underset{Q}{\operatorname{argmax}} P(Q|O) = \underset{Q}{\operatorname{argmax}} \frac{P(Q,O)}{P(O)} = \underset{Q}{\operatorname{argmax}} P(Q,O).$$

   On Tuesday, we discussed how to find Q^* using the Viterbi algorithm. This process is called "decoding" because we decode the sequence of symbols to determine the hidden sequence of states. In speech recognition, recorded speech is "decoded" into words or phonemes to determine the meaning of the utterance.

2. The total probability of O is the sum of the probability of O over all possible paths through the HMM:

$$P(O) = \sum_{Q} P(O|Q)P(Q) = \sum_{Q} P(O,Q)$$

To solve this we use the Forward algorithm, which interatively calculates the probability of being in state S_i after generating the sequence up to observation O_t. We designate this quantity:

$$\alpha_t(i) = P(O_1, O_2, O_3, ...O_t, q_t = S_i)$$

Algorithm: Forward

Initialization:

$$\alpha_1(i) = \pi_i e_i(O_1)$$

Iteration:

$$\alpha_t(i) = \sum_{j=1}^{N} \alpha_{t-1}(j) * a_{ji} * e_i(O_t)$$

The probability of observing the entire sequence is given by the sum over all possible final states:

$$P(O) = \sum_{i=1}^{N} \alpha_T(i)$$

3. We wish to determine $P(q_t = S_i|O_t)$, the probability of being in state S_i when O_t is emitted. This is equivalent to the emitting $O_1 \ldots O_{t-1}$ over any path, entering S_i, emitting O_t, and then emitting $O_{t+1} \ldots O_T$ over any path or

$$P(q_t = S_i|O_t) = P(O_1, O_2, O_3, ...O_t, q_t = S_i)P(O_{t+1}, O_{t+2}, ...O_T|q_t = S_i)$$

Note that the first term is just $\alpha_t(i)$. We calculate the second term with the *Backward algorithm*, described in the next section. The Backward algorithm can also be used instead of the Forward algorithm to calculate $P(O)$ and it is used in the Baum-Welch algorithm for estimating the parameters of the model.

2.2 Backward algorithm

Above we calculated forward in time: $\alpha_1(\cdot), \alpha_2(,\cdot), ..., \alpha_T(,\cdot)$. It is also possible to calculate the probability of emmitting O given a particular model by working backward from the end of the sequence. Let $\beta_t(i)$ be the probability of generating the last $T-t+1$ observations given that O_{t-1} was emitted from state i:

$$\beta_t(i) = P(O_t, O_{t+1}, O_{t+2}, ...O_T | q_{t-1} = S_i)$$

Algorithm: Backward

Initialization:

$$\beta_T(i) = \sum_{j=1}^{N} a_{ij} * e_j(O_T)$$

Iteration:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} * e_j(O_t) * \beta_{t+1}(j)$$

To determine $P(q_t = S_i | O_t)$, we work backwards from O_T to O_{t+1}, to obtain

$$P(q_t = S_i | O_t) = \alpha_t(i) \cdot \beta_{t+1}(i)$$

To calculate the probability of the entire sequence, we start with T and go all the way to 2, calculating $\beta_t(i)$.

$$P(O) = \sum_{j=1}^{N} \pi_j e_j(O_1) \beta_2(j)$$

Determining the most probable state for every symbol O_t is an alternative to Viterbi decoding. This could give better results in some cases, such as when suboptimal paths are almost as probable as the most probable path.

Either the Forward or the Backward algorithm can be used to determine the probability of a sequence, O. Both are needed in order to learn parameters from unlabeled data using the Baum Welch procedure.