Applying HMMs to IE

- **Document** $\Rightarrow$ generated by a stochastic process modelled by an HMM
- **Token** $\Rightarrow$ word
- **State** $\Rightarrow$ “reason/explanation” for a given token
  - ‘Background’ state emits tokens like ‘the’, ‘said’, …
  - ‘Money’ state emits tokens like ‘million’, ‘euro’, …
  - ‘Organization’ state emits tokens like ‘university’, ‘company’, …
- **Extraction**: via the Viterbi algorithm, a dynamic programming technique for efficiently computing the most likely sequence of states that generated a document.
What is an HMM?

- Graphical Model Representation: Variables by time
- Circles indicate states
- Arrows indicate probabilistic dependencies between states
What is an HMM?

- Green circles are *hidden states*
- Dependent only on the previous state: Markov process
- “The past is independent of the future given the present.”
What is an HMM?

- Purple nodes are *observed states*
- Dependent only on their corresponding hidden state
HMM Formalism

- \{S, K, \Pi, A, B\}
- \(S : \{s_1 \ldots s_N\}\) are the values for the hidden states
- \(K : \{k_1 \ldots k_M\}\) are the values for the observations
HMM Formalism

- \{S, K, \Pi, A, B\}
- \Pi = \{\pi_t\} are the initial state probabilities
- A = \{a_{ij}\} are the state transition probabilities
- B = \{b_{ik}\} are the observation state probabilities
Inference for an HMM

- Compute the probability of a given observation sequence
- Given an observation sequence, compute the most likely hidden state sequence
- Given an observation sequence and set of possible models, which model most closely fits the data?
Sequence Probability

Given an observation sequence and a model, compute the probability of the observation sequence

\[ O = (o_1, \ldots, o_T), \mu = (A, B, \Pi) \]

Compute \( P(O \mid \mu) \)
Sequence probability

\[ P(O \mid X, \mu) = b_{x_1 o_1} b_{x_2 o_2} \ldots b_{x_T o_T} \]

\[ P(X \mid \mu) = \pi_{x_1} a_{x_1 x_2} a_{x_2 x_3} \ldots a_{x_{T-1} x_T} \]

\[ P(O, X \mid \mu) = P(O \mid X, \mu) P(X \mid \mu) \]

\[ P(O \mid \mu) = \sum_X P(O \mid X, \mu) P(X \mid \mu) \]
Sequence probability

\[
P(O | \mu) = \sum_{\{x_1 \ldots x_T\}} \prod_{t=1}^{T-1} a_{x_t x_{t+1}} b_{x_{t+1} o_t} \prod_{t=1}^{T-1} b_{x_t o_t}
\]
• Special structure gives us an efficient solution using dynamic programming.

• **Intuition:** Probability of the first $t$ observations is the same for all possible $t + 1$ length state sequences.

• Define: $\alpha_i(t) = P(o_1...o_t, x_t = i | \mu)$
Forward Procedure

\[ \alpha_j(t + 1) = P(o_1 \ldots o_{t+1}, x_{t+1} = j) \]

\[ = P(o_1 \ldots o_{t+1} \mid x_{t+1} = j)P(x_{t+1} = j) \]

\[ = P(o_1 \ldots o_t \mid x_{t+1} = j)P(o_{t+1} \mid x_{t+1} = j)P(x_{t+1} = j) \]

\[ = P(o_1 \ldots o_t, x_{t+1} = j)P(o_{t+1} \mid x_{t+1} = j) \]
Forward Procedure

\[
\sum_{i=1 \ldots N} P(o_1 \ldots o_t, x_t = i, x_{t+1} = j) P(o_{t+1} | x_{t+1} = j)
\]

\[
= \sum_{i=1 \ldots N} P(o_1 \ldots o_t, x_{t+1} = j | x_t = i) P(x_t = i) P(o_{t+1} | x_{t+1} = j)
\]

\[
= \sum_{i=1 \ldots N} P(o_1 \ldots o_t, x_t = i) P(x_{t+1} = j | x_t = i) P(o_{t+1} | x_{t+1} = j)
\]

\[
= \sum_{i=1 \ldots N} \alpha_i(t) a_{ij} b_{jo_{t+1}}
\]
Backward Procedure

\[ \beta_i(T + 1) = 1 \]

\[ \beta_i(t) = P(o_t \ldots o_T \mid x_t = i) \]

\[ \beta_i(t) = \sum_{j=1}^{N} a_{ij} b_{io} \beta_j(t + 1) \]

Probability of the rest of the states given the first state
Sequence probability

\[
P(O \mid \mu) = \sum_{i=1}^{N} \alpha_i(T)
\]
Forward Procedure

\[
P(O \mid \mu) = \sum_{i=1}^{N} \pi_i \beta_i(1)
\]
Backward Procedure

\[
P(O \mid \mu) = \sum_{i=1}^{N} \alpha_i(t) \beta_i(t)
\]
Combination
Best State Sequence

- Find the state sequence that best explains the observations
- Viterbi algorithm
- \( \arg \max_X P(X \mid O) \)
Viterbi Algorithm

\[ \delta_j(t) = \max_{x_1...x_{t-1}, o_1...o_{t-1}, x_t = j, o_t} P(x_1...x_{t-1}, o_1...o_{t-1}, x_t = j, o_t) \]

The state sequence which maximizes the probability of seeing the observations to time \( t-1 \), landing in state \( j \), and seeing the observation at time \( t \)
Viterbi Algorithm

\[ \delta_j(t) = \max_{x_1 \ldots x_{t-1}} P(x_1 \ldots x_{t-1}, o_1 \ldots o_{t-1}, x_t = j, o_t) \]

\[ \delta_j(t + 1) = \max_i \delta_i(t) a_{ij} b_{jo_{t+1}} \]

\[ \psi_j(t + 1) = \arg \max_i \delta_i(t) a_{ij} b_{jo_{t+1}} \]
Best State Sequence: Viterbi Algorithm, Trellis View

\[ \delta_j(t + 1) = \max_i \delta_i(t) a_{ij} b_{jo_{t+1}} \]

\[ \psi_j(t + 1) = \arg \max_i \delta_i(t) a_{ij} b_{jo_{t+1}} \]

Find biggest at last time, and then trace backwards.
Compute the most likely state sequence by working backwards.

\[ \hat{X}_T = \arg \max_i \delta_i (T) \]
\[ \hat{X}_t = \psi_{\hat{X}_{t+1}} (t + 1) \]
\[ P(\hat{X}) = \arg \max_i \delta_i (T) \]
Learning = Parameter Estimation: EM/Forward-Backward algorithm

• Given an observation sequence, find the model that is most likely to produce that sequence.
  • Find parameters so $P(O|\Theta)$ is maximized
• No analytic method, so:
  • Given a model and observation sequence, update the model parameters to better fit the observations: hill climb so $P(O|\Theta)$ goes up.
Parameter Estimation: Baum-Welch or Forward-Backward

\[ p_t(i, j) = \frac{\alpha_i(t) a_{ij} b_{j \omega_{t+1}} \beta_j(t+1)}{\sum_{m=1}^{N} \alpha_m(t) \beta_m(t)} \]

- Probability of traversing an arc

\[ \gamma_i(t) = \sum_{j=1}^{N} p_t(i,j) \]

- Probability of being in state \( i \)
Parameter Estimation: Baum-Welch or Forward-Backward

Now we can compute the new estimates of the model parameters.

\[ \hat{a}_{ij} = \frac{\sum_{t=1}^{T} p_t(i, j)}{\sum_{t=1}^{T} \gamma_i(t)} \]

\[ \hat{b}_{ik} = \frac{\sum_{t:o_t=k} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_i(t)} \]

\[ \hat{\pi}_i = \gamma_i(1) \]
References