Hidden Markov models
First, let’s review

• **Zipf** in the context of *entropy*.
• Intuitions about **KL divergence**.
• Reminder of **statistical significance**.
• **Collocations vs. idioms**.
Zipf’s law in the context of entropy
Zipf’s law in the context of entropy

- From this perspective, it is clear that a small number of highly-ranked words have a fairly low entropy.
Zipf’s law in the context of entropy

• However, among the many less frequent words, entropy quickly rises.

• About 50% of the words in large corpora, like Brown, occur only once.

• This implies difficulty in ‘preferring’ certain words over others.

• This is related to sparseness: each hapax legomenon means $2(||\mathcal{V}|| - 1)$ zeros in our unsmoothed bigram table.
Kullback-Leibler divergence

- E.g., I have two distributions: \( P \) (learned from *Shakespeare*) and \( Q \) (learned from *Wall Street Journal*).

\[
D_{KL}(P||Q) = \sum_{w} P(w) \log \frac{P(w)}{Q(w)}
\]
KL divergence and equivocation

\[ D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \]

\[ H(Y|X) = \sum_{x \in X} p(x) H(Y|X = x) \]

- \( D_{KL} \) tells us the expected number of extra bits required to code samples from \( P \) when using a code based on \( Q \).

- It is not the same as conditional entropy (aka equivocation).
Kullback-Leibler divergence

- Computed on a per-word basis. Some words may be much more likely in one distribution than the other.

\[ D_{KL}(P||Q) = \sum_w P(w) \log \frac{P(w)}{Q(w)} = \cdots + P(OPEC) \log \frac{P(OPEC)}{Q(OPEC)} + \cdots \]
Kullback-Leibler divergence

\[ D_{KL}(P||Q) = \cdots + P(OPEC) \log \frac{P(OPEC)}{Q(OPEC)} + \cdots \]

- As \( P(OPEC) \to Q(OPEC) \), then \( \left( \log \frac{P(OPEC)}{Q(OPEC)} \right) \to 0 \). There is little to no divergence due to this word.

- As \( P(OPEC) \to 0 \), then \( \left( P(OPEC) \log \frac{P(OPEC)}{Q(OPEC)} \right) \to 0 \). We never expect to have to encode \( OPEC \) in \( P \), so there is no divergence.

- We are constrained that \( Q(w) > 0 \) for every \( w \) s.t \( P(w) > 0 \).
Statistical significance

• The **purpose** of hypothesis testing is to show that the difference between distribution means is **not** due to chance.

  • Saying system $A$ is better than system $B$ is **inadequate**.
  • Saying $A$ is better than $B$ at 99% level of confidence implies that $A$ is **so** much better, we expect it to win in 99% of all possible tests.
Idioms ≠ collocations

• **Collocation:** *n.* a ‘turn-of-phrase’ or usage where a sequence of words is perceived to have a meaning ‘beyond’ the sum of its parts.

• **Idiom:** *n.* An expression with a **figurative** meaning **unrelated** to the literal meanings of its words.
  • E.g.,

<table>
<thead>
<tr>
<th>Idiom</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spill the beans</td>
<td>Divulge a secret</td>
</tr>
<tr>
<td>Burn the midnight oil</td>
<td>Work late</td>
</tr>
<tr>
<td>Under the weather</td>
<td>Somewhat ill</td>
</tr>
<tr>
<td>Get out of hand</td>
<td>Become uncontrollable</td>
</tr>
</tbody>
</table>
Collocations

• Note that collocations ≠ idioms.

• Idioms are ‘non-compositional’ in that their meanings are not in any way derived from their components.

• Collocations have meanings that are derived from their component words.
Idioms

• **Idiom**:  *n.* An expression with a **figurative** meaning **unrelated** to the literal meanings of its words.

• Like collocations, you cannot substitute near-synonyms, e.g.:
  • *Spill the beans*:  *pour some beans*
  • *Burn the midnight oil*:  *ignite the midnight petroleum*
  • *Under the weather*:  *beneath the weather*
  • *Get out of hand*:  *leave the hand*

• Although you *may* sometimes derive meanings of **idioms** from historical context, they remain highly **metaphorical**.
Collocations

• E.g., *soft drink* is a collocation because in practice you will not see substitutions of either word
  • You would not say *mild drink* or *soft beverage*.
  • However, it *is* a drink – the meaning of the phrase is related to its parts.

• Similarly, *disk drive* and *video recorder* are relatively immutable, but their meanings *are related* to their component words.
Week 4

• (Hidden) Markov models.
  • Using them.
  • Training them.
  • Loving them.
Observable Markov model

• We’ve seen this type of model:
  • e.g., consider the 7-word vocabulary:
    \{ship, pass, camp, frock, soccer, mother, tops\}

• What is the probability of the sequence
  ship, ship, pass, ship, tops

• Assuming a \textbf{bigram} model (i.e., 1^{st}\text{-order Markov}),
  \[ P(\text{ship}|<s>)P(\text{ship}|\text{ship})P(\text{pass}|\text{ship}) \]
  \[ \cdot P(\text{ship}|\text{pass})P(\text{tops}|\text{ship}) \]
Observable Markov model

• This can be conceptualized graphically.

• We start with $N$ states, $s_1, s_2, \ldots, s_N$ that represent unique observations in the world.

• Here, $N = 7$ and each state represents one of the words we can observe.
Observable Markov model

- We have discrete **timesteps**, \( t = 0, t = 1, \ldots \)
- On the \( t^{th} \) timestep the system is in exactly one of the available states, \( q_t \).
  - \( q_t \in \{s_1, s_2, \ldots, s_N\} \)
- We could start in any state. The probability of starting with a particular state \( s \) is \( P(q_0 = s) = \pi(s) \)
Observable Markov model

- At each step we must move to a state with some probability.

- Here, an arrow from $q_t$ to $q_{t+1}$ represents $P(q_{t+1}|q_t)$

- $P(\text{ship}|\text{ship})$
- $P(\text{tops}|\text{ship})$
- $P(\text{pass}|\text{ship})$
- $P(\text{frock}|\text{ship}) = 0$
Observable Markov model

- Probabilities on all outgoing arcs must sum to 1.

\[ P(\text{ship}|\text{ship}) + P(\text{tops}|\text{ship}) + P(\text{pass}|\text{ship}) = 1 \]

\[ P(\text{ship}|\text{tops}) + P(\text{tops}|\text{tops}) + P(\text{mother}|\text{tops}) = 1 \]

- ...
A multivariate system

- What if the probabilities of observing words depended *only* on some *other* variable, like *mood*?

<table>
<thead>
<tr>
<th>word</th>
<th>P(word)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>0.1</td>
</tr>
<tr>
<td>pass</td>
<td>0.05</td>
</tr>
<tr>
<td>camp</td>
<td>0.05</td>
</tr>
<tr>
<td>frock</td>
<td>0.6</td>
</tr>
<tr>
<td>soccer</td>
<td>0.05</td>
</tr>
<tr>
<td>mother</td>
<td>0.1</td>
</tr>
<tr>
<td>tops</td>
<td>0.05</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>word</th>
<th>P(word)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>0.25</td>
</tr>
<tr>
<td>pass</td>
<td>0.25</td>
</tr>
<tr>
<td>camp</td>
<td>0.05</td>
</tr>
<tr>
<td>frock</td>
<td>0.3</td>
</tr>
<tr>
<td>soccer</td>
<td>0.05</td>
</tr>
<tr>
<td>mother</td>
<td>0.09</td>
</tr>
<tr>
<td>tops</td>
<td>0.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>word</th>
<th>P(word)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>0.3</td>
</tr>
<tr>
<td>pass</td>
<td>0.0</td>
</tr>
<tr>
<td>camp</td>
<td>0.0</td>
</tr>
<tr>
<td>frock</td>
<td>0.2</td>
</tr>
<tr>
<td>soccer</td>
<td>0.05</td>
</tr>
<tr>
<td>mother</td>
<td>0.05</td>
</tr>
<tr>
<td>tops</td>
<td>0.4</td>
</tr>
</tbody>
</table>
A multivariate system

• What if that variable **changes** over time?
  • e.g., I’m **happy** one second and **disgusted** the next.
• Here, **state** $\equiv$ mood
  **observation** $\equiv$ word.
Observable multivariate systems

• Imagine you have access to my emotional state somehow.

• All your data are completely **observable** at every timestep.

• E.g.,

<table>
<thead>
<tr>
<th>$t$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>😊</td>
<td>😊</td>
<td>😊</td>
<td>...</td>
</tr>
<tr>
<td>word</td>
<td>mother</td>
<td>flock</td>
<td>soccer</td>
<td>...</td>
</tr>
</tbody>
</table>

$\equiv$

$\langle \text{mother, flock, soccer}, \langle 😊, 😊, 😊 \rangle \rangle$
Observable multivariate systems

- What is the probability of a sequence of words and states?
  
  $P(w_{0:t}, q_{0:t}) = P(q_{0:t})P(w_{0:t} | q_{0:t}) \approx \prod_{i=0}^{t} P(q_i | q_{i-1})P(w_i | q_i)$

- e.g.,
  
  $P(\langle ship, pass \rangle, \langle :)\), :)\rangle) = P(q_0 = :)P(ship | :)P(\circ| :)P(pass | :)$
Observable multivariate systems

**Q:** How do you **learn** these probabilities?

\[ P(w_{0:t}, q_{0:t}) \approx \prod_{i=0}^{t} P(q_i | q_{i-1}) P(w_i | q_i) \]

**A:** When all data are observed, basically the same as before.

\[ P(q_i | q_{i-1}) = \frac{P(q_{i-1}q_i)}{P(q_{i-1})} \] is learned with MLE from training data.

\[ P(w_i | q_i) = \frac{P(w_i,q_i)}{P(q_i)} \] is also learned with MLE from training data.
Hidden variables

• Q: What if you don’t know the states during testing?
  • e.g., compute $P((\text{ship, ship, pass, frock}))$

• Q: What if you don’t know the states during training?
Examples of hidden phenomena

• We want to represent **surface** (i.e., **observable**)
  phenomena as the **output** of **hidden** underlying systems.
  
  • e.g.,
    • **Words** are the outputs of hidden **parts-of-speech**,
    • **French phrases** are the outputs of hidden **English phrases**,
    • **Speech sounds** are the outputs of hidden **phonemes**.

• in other fields,
  • **Encrypted symbols** are the outputs of hidden **messages**,
  • **Genes** are the outputs of hidden **functional relationships**,
  • **Weather** is the output of hidden **climate conditions**,
  • **Stock prices** are the outputs of hidden **market conditions**,
  • ...
Definition of an HMM

- A hidden Markov model (HMM) is specified by the 5-tuple \( \{S, W, \Pi, A, B\} \):
  - \( S = \{s_1, ..., s_N\} \): set of states (e.g., moods)
  - \( W = \{w_1, ..., w_K\} \): output alphabet (e.g., words)
  - \( \Pi = \{\pi_1, ..., \pi_N\} \): initial state probabilities
  - \( A = \{a_{ij}\}, i, j \in S \): state transition probabilities
  - \( B = b_i(w), i \in S, w \in W \): state output probabilities

yielding
  - \( Q = \{q_0, ..., q_T\}, q_i \in S \): state sequence
  - \( O = \{\sigma_0, ..., \sigma_T\}, \sigma_i \in W \): output sequence
A hidden Markov production process

- An HMM is a representation of a process in the world.
  - We can synthesize data, as in Shannon’s game.
- This is how an HMM generates new sequences:
  - \( t := 0 \)
  - **Start** in state \( q_0 = s_i \) with probability \( \pi_i \)
  - **Emit** observation symbol \( \sigma_0 = w_k \) with probability \( b_i(\sigma_0) \)
  - **While** (not forever)
    - **Go** from state \( q_t = s_i \) to state \( q_{t+1} = s_j \) with probability \( a_{ij} \)
    - **Emit** observation symbol \( \sigma_{t+1} = w_k \) with probability \( b_j(\sigma_{t+1}) \)
    - \( t := t + 1 \)
Fundamental tasks for HMMs

1. Given a model with particular parameters \( \theta = \langle \Pi, A, B \rangle \), how do we efficiently compute the likelihood of a particular observation sequence, \( P(O; \theta) \)?

We previously computed the probabilities of word sequences using \( N \)-grams.

The probability of a particular sequence is usually useful as a means to some other end.
2. Given an observation sequence $\mathcal{O}$ and a model $\theta$, how do we choose a state sequence $Q = \{q_0, \ldots, q_T\}$ that best explains the observations?

This is the task of inference – i.e., guessing at the best explanation of unknown (‘latent’) variables given our model.

This is often an important part of classification.
3. Given a large observation sequence $\mathcal{O}$, how do we choose the best parameters $\theta = \langle \Pi, A, B \rangle$ that explain the data $\mathcal{O}$?

This is the task of \textit{training}.

As before, we want our parameters to be set so that the available training data is maximally likely, But doing so will involve guessing unseen information.
A pro golfer can only putt the ball 3, 5, 7, and 11 meters.

He is currently 20m from the hole.

If he only sinks the ball if it stops directly in the hole, what is the minimum number of strokes to sink the ball?
Answer: Golfer

- It takes two strokes if the golfer uses >1 dimensions.

Lesson: this is the kind of question asked in job interviews
Task 1: Computing $P(\mathcal{O}; \theta)$

• We’ve seen the probability of a joint sequence of observations and states:

$$P(\mathcal{O}, Q; \theta) = P(\mathcal{O}|Q; \theta)P(Q; \theta)$$

$$= \pi_{q_0} b_{q_0}(\sigma_0) a_{q_0q_1} b_{q_1}(\sigma_1) a_{q_1q_2} b_{q_2}(\sigma_2) \ldots$$

• To get the probability of our observations without seeing the state, we must sum over all possible state sequences:

$$P(\mathcal{O}; \theta) = \sum_{Q} P(\mathcal{O}|Q; \theta)P(Q; \theta).$$
Computing $P(O; \theta)$ na"ively

• To get the total probability of our observations, we could directly sum over all possible state sequences:

$$P(O; \theta) = \sum_Q P(O|Q; \theta)P(Q; \theta).$$

• For observations of length $T$, each state sequence involves $2T$ multiplications (1 for each state transition, 1 for each observation, 1 for the start state, minus 1).

• There are up to $N^T$ possible state sequences of length $T$ given $N$ states.

$$\therefore \sim (1 + T + T - 1) \cdot N^T$$ multiplications
Computing $P(\mathcal{O}; \theta)$ cleverly

• To avoid this complexity, we use **dynamic programming**; we **remember**, rather than **recompute**, partial results.

• We make a **trellis** which is an array of **states vs. time**.
  • The element at $(i, t)$ is $\alpha_i(t)$
    the probability of being in state $i$ at time $t$
    **after seeing all previous observations**:
    $P(\sigma_{o:t-1}, q_t = s_i; \theta)$
Trellis

Probability of being in state $s_3$ at time $t = 2$
Alternative paths through the trellis

State

Probability of being in state $s_3$ at time $t = 2$
Alternative paths through the trellis

State

$S_1$

$S_2$

$S_3$

$S_N$

0

1

2

$T - 1$

Time, $t$

Probability of being in state $s_3$ at time $t = 2$
Alternative paths through the trellis

State

<table>
<thead>
<tr>
<th>$s_1$</th>
<th>$s_2$</th>
<th>$s_3$</th>
<th>$s_N$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>$T-1$</td>
</tr>
</tbody>
</table>

Time, $t$

- Probability of being in state $s_3$ at time $t = 2$

Rf012

Rc
d

Probability of being in state $s_3$ at time $t = 2$
Alternative paths through the trellis

Notice that I already computed a path through this node

Probability of being in state $s_3$ at time $t = 2$
Alternative paths through the trellis

Notice that I already computed a path through this node

Probability of being in state $s_3$ at time $t = 2$
AND SO ON...
To compute the probabilities of the black node and the yellow node, I need (among others) the probabilities of the orange node and the purple node:

I compute once, and save them.
The Forward procedure

• To compute

\[ \alpha_i(t) = P(\sigma_{0:t}, q_t = s_i; \theta) \]

we can compute \( \alpha_j(t - 1) \) for possible previous states \( s_j \), then use our knowledge of \( a_{ji} \) and \( b_i(\sigma_t) \)

• We compute the trellis left-to-right (because of time) and top-to-bottom (because of convention).

• Remember: \( \sigma_t \) is fixed and known. \( \alpha_i(t) \) is agnostic of the future.
The Forward procedure

• The trellis is computed left-to-right and top-to-bottom.

• There are three steps in this procedure:
  • Initialization: Compute the nodes in the first column of the trellis \( t = 0 \).
  • Induction: Iteratively compute the nodes in the rest of the trellis \( 1 \leq t < T \).
  • Conclusion: Sum over the nodes in the last column of the trellis \( t = T - 1 \).
Initialization of Forward procedure

\[ \alpha_i(0) := \pi_i b_i(\sigma_0), \quad i := 1..N \]

(Probability of starting in state \(i\) and reading the first word there)
Induction of Forward procedure

\[
\alpha_j(t + 1) := \sum_{i=1}^{N} \alpha_i(t) a_{ij} b_j(\sigma_{t+1}),
\]

for \( j := 1..N, t := 0..(T - 2) \)

(Probability of getting to state \( j \) at time \( t + 1 \))
Induction of Forward procedure

\[ s_1 \quad \alpha_1(t) \]
\[ s_2 \quad \alpha_2(t) \]
\[ s_3 \quad \alpha_3(t) \]
\[ s_N \quad \alpha_N(t) \]

\[ s_j \quad \alpha_j(t + 1) \]

\[ a_{1j}b_j(\sigma_{t+1}) \]
\[ a_{2j}b_j(\sigma_{t+1}) \]
\[ a_{3j}b_j(\sigma_{t+1}) \]
\[ a_{Nj}b_j(\sigma_{t+1}) \]

\[ t \]
\[ t + 1 \]
Conclusion of Forward procedure

\[ P(0; \theta) = \sum_{i=1}^{N} \alpha_i(T - 1) \]

Sum over all possible final states.
The Forward procedure

• The naïve approach needed \((2T) \cdot N^T\) multiplications.

• The Forward procedure (using **dynamic programming**) needs only \(2N^2T\) multiplications. 😊

• The Forward procedure gives us \(P(\emptyset; \theta)\).

• Clearly, but less intuitively, we can also compute the trellis from back-to-front, i.e., backwards in time...
The Backward procedure

• In the \((i, t)^{th}\) node of the trellis, we store

\[
\beta_i(t) = P(\sigma_{t+1:T} | q_t = s_i; \theta)
\]

which is computed by summing probabilities on outgoing arcs from that node.

\(\beta_i(t)\) is the probability of starting in state \(i\) at time \(t\) then observing everything that comes thereafter.

• The trellis is computed right-to-left and top-to-bottom.
The Backward procedure

• Initialization
  \[ \beta_i(T - 1) = 1, \quad i := 1..N \]

• Induction
  \[ \beta_i(t) = \sum_{j=1}^{N} a_{ij} b_j(\sigma_{t+1}) \beta_j(t + 1), \quad i := 1..N \]

• Conclusion
  \[ P(\mathcal{O}; \theta) = \sum_{i=1}^{N} \pi_i b_i(\sigma_0) \beta_i(0) \]
The Backward procedure – so what?

• The **combination** of Forward and Backward procedures will be vital for solving parameter re-estimation, i.e., **training**.

• Generally, we can **combine** $\alpha$ and $\beta$ at any point in time to represent the probability of an **entire** observation sequence:

(Next slide, please)
Combining $\alpha$ and $\beta$

$$P(O, q_t = i; \theta) = \alpha_i(t)\beta_i(t)$$

$$\therefore P(O; \theta) = \sum_{i=1}^{N} \alpha_i(t)\beta_i(t)$$

This is not just fun trivia – we’ll see soon why it can be useful...
Example – PoS state sequences

- Will/MD the/DT chair/NN chair/?? the/DT meeting/NN from/IN that/DT chair/NN?

a) MD → DT → NN → VB → ...
   Will → the → chair → chair

b) MD → DT → NN → NN → ...
   Will → the → chair → chair
Task 2: Choosing $Q = \{q_0 \ldots q_T\}$

- The purpose of finding the best state sequence $Q^*$ out of all possible state sequences $Q$ is that it tells us what is most likely to be going on ‘under the hood’.
  - E.g., it tells us the most likely part-of-speech tags,
  - E.g., it tells us the most likely English words given French translations (*in a very simple model).

- With the Forward algorithm, we didn’t care about specific state sequences – we were summing over all possible state sequences.
Task 2: Choosing $Q = \{q_0 \ldots q_T\}$

• In other words,

$$Q^* = \arg\max_Q P(\mathcal{O}, Q; \theta)$$

where

$$P(\mathcal{O}, Q; \theta) = \pi_{q_0} b_{q_0}(\sigma_0) \prod_{t=1}^{T} \alpha_{q_{t-1}q_t} b_{q_t}(\sigma_t)$$
Recall

- Observation likelihoods depend on the state, which changes over time

- We **cannot** simply choose the state that maximizes the probability of $o_t$ without considering the state sequence.

<table>
<thead>
<tr>
<th>word</th>
<th>$P(word)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>0.25</td>
</tr>
<tr>
<td>pass</td>
<td>0.25</td>
</tr>
<tr>
<td>camp</td>
<td>0.05</td>
</tr>
<tr>
<td>flock</td>
<td>0.3</td>
</tr>
<tr>
<td>soccer</td>
<td>0.05</td>
</tr>
<tr>
<td>mother</td>
<td>0.09</td>
</tr>
<tr>
<td>tops</td>
<td>0.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>word</th>
<th>$P(word)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>0.3</td>
</tr>
<tr>
<td>pass</td>
<td>0.0</td>
</tr>
<tr>
<td>camp</td>
<td>0.0</td>
</tr>
<tr>
<td>flock</td>
<td>0.2</td>
</tr>
<tr>
<td>soccer</td>
<td>0.05</td>
</tr>
<tr>
<td>mother</td>
<td>0.05</td>
</tr>
<tr>
<td>tops</td>
<td>0.4</td>
</tr>
</tbody>
</table>
The Viterbi algorithm

• The Viterbi algorithm is an inductive dynamic-programming algorithm that uses a new kind of trellis.

• We define the probability of the most probable path leading to the trellis node at (state \(i\), time \(t\)) as

\[
\delta_i(t) = \max_{q_0 \ldots q_{t-1}} P(q_0 \ldots q_{t-1}, \sigma_0 \ldots \sigma_{t-1}, q_t = s_i; \theta)
\]

• \(\psi_i(t)\): The best possible previous state, if I’m in state \(i\) at time \(t\).
**Viterbi example**

- For illustration, we assume a simpler state-transition topology:

```
<table>
<thead>
<tr>
<th>word</th>
<th>P(word)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>0.25</td>
</tr>
<tr>
<td>pass</td>
<td>0.25</td>
</tr>
<tr>
<td>camp</td>
<td>0.05</td>
</tr>
<tr>
<td>flock</td>
<td>0.3</td>
</tr>
<tr>
<td>soccer</td>
<td>0.05</td>
</tr>
<tr>
<td>mother</td>
<td>0.09</td>
</tr>
<tr>
<td>tops</td>
<td>0.01</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>word</th>
<th>P(word)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>0.1</td>
</tr>
<tr>
<td>pass</td>
<td>0.05</td>
</tr>
<tr>
<td>camp</td>
<td>0.05</td>
</tr>
<tr>
<td>flock</td>
<td>0.6</td>
</tr>
<tr>
<td>soccer</td>
<td>0.05</td>
</tr>
<tr>
<td>mother</td>
<td>0.1</td>
</tr>
<tr>
<td>tops</td>
<td>0.05</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>word</th>
<th>P(word)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ship</td>
<td>0.3</td>
</tr>
<tr>
<td>pass</td>
<td>0</td>
</tr>
<tr>
<td>camp</td>
<td>0</td>
</tr>
<tr>
<td>flock</td>
<td>0.2</td>
</tr>
<tr>
<td>soccer</td>
<td>0.05</td>
</tr>
<tr>
<td>mother</td>
<td>0.05</td>
</tr>
<tr>
<td>tops</td>
<td>0.4</td>
</tr>
</tbody>
</table>
```
Step 1: Initialization of Viterbi

- Initialize with $\delta_0(i) = \pi_i b_i(\sigma_0)$ and $\psi_i(0) = 0$ for all states.
Step 1: Initialization of Viterbi

- For example, let’s assume
  \[ \pi_d = 0.8, \pi_h = 0.2 \quad \text{and} \quad \mathcal{O} = \text{ship, frock, tops} \]

<table>
<thead>
<tr>
<th>Probability</th>
<th>Observation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 \cdot 0.25</td>
<td>ship</td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0.2 \cdot 0.3</td>
<td>frock</td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0.8 \cdot 0.1</td>
<td>tops</td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

\( \delta \): max probability
\( \psi \): backtrace

Observations, \( \sigma_t \):
- \( \sigma_0 = \text{ship} \)
- \( \sigma_1 = \text{frock} \)
- \( \sigma_2 = \text{tops} \)
Step 2: Induction of Viterbi

The best path to state $s_j$ at time $t$, $\delta_j(t)$, depends on the best path to each possible previous state, $\delta_i(t - 1)$, and their transitions to $j$, $a_{ij}$

$$\delta_j(t) = \max_i [\delta_i(t - 1)a_{ij}] b_j(\sigma_t)$$

$$\psi_j(t) = \arg\max_i [\delta_i(t - 1)a_{ij}]$$

$\sigma_0 = ship$

$\sigma_1 = frock$

$\sigma_2 = tops$

Observations, $\sigma_t$
Step 2: Induction of Viterbi

Specifically...

<table>
<thead>
<tr>
<th>Observation, $\sigma_t$</th>
<th>$\delta_s(1) = \max_i [\delta_i(0)a_{is}] b_s(\sigma_1)$</th>
<th>$\psi_s(1) = \arg\max_i [\delta_i(0)a_{is}]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma_0 = \text{ship}$</td>
<td>$0$</td>
<td>$0$</td>
</tr>
<tr>
<td>$\sigma_1 = \text{frock}$</td>
<td>$0.06$</td>
<td>$0$</td>
</tr>
<tr>
<td>$\sigma_2 = \text{tops}$</td>
<td>$0.08$</td>
<td>$0$</td>
</tr>
</tbody>
</table>

Observation cells:
Step 2: Induction of Viterbi

\[
\delta_s(0) = 0, a_{sd} = 0, \quad \therefore \delta_s(0)a_{sd} = 0
\]

\[
\delta_h(0) = 0.06, a_{hd} = 0, \quad \therefore \delta_h(0)a_{hd} = 0
\]

\[
\delta_d(0) = 0.08, a_{dd} = 0.4, \quad \therefore \delta_d(0)a_{dd} = 0.032
\]

\[
\sigma_0 = \text{ship}
\]

\[
\sigma_1 = \text{frock}
\]

\[
\sigma_2 = \text{tops}
\]

Observations, \(\sigma_t\)
Step 2: Induction of Viterbi

\[ \delta_s(1) = \max_i [\delta_i(0)a_{is}] b_s(o_1) \]

\[ \delta_d(0)a_{dd} = 0.032, \quad b_d(frock) = 0.6 \]

\[ \therefore \max_i [\delta_i(0)a_{id}] b_d(o_1) = 1.92 \times 10^{-2} = 1.92E^{-2} \]

\[ \phi_h(1) = \arg\max_i [\delta_i(0)a_{ih}] \]

\[ d \text{ was the most likely previous state} \]

\[ \sigma_0 = \text{ship} \quad \sigma_1 = \text{frock} \quad \sigma_2 = \text{tops} \]

Observations, \( \sigma_t \)
Step 2: Induction of Viterbi

\[
\begin{align*}
\delta_s(0) &= 0, a_{sh} = 0, & \therefore \delta_s(0)a_{sh} &= 0 \\
\delta_h(0) &= 0.06, a_{hh} = 0.8, & \therefore \delta_h(0)a_{hh} &= 0.048 \\
\delta_d(0) &= 0.08, a_{dh} = 0.5, & \therefore \delta_d(0)a_{dh} &= 0.04 \\
\end{align*}
\]

\[
\begin{align*}
\max_i [\delta_i(0)a_{ih}] b_h(\sigma_1) \\
\arg\max_i [\delta_i(0)a_{ih}] \\
\end{align*}
\]

Observations, \(\sigma_t\)

\(\sigma_0 = ship\) \hspace{1cm} \(\sigma_1 = frock\) \hspace{1cm} \(\sigma_2 = tops\)
Step 2: Induction of Viterbi

\[
\delta_h(0)a_{hh} = 0.048, \quad b_h(frock) = 0.2
\]

\[
\therefore \max_i [\delta_i(0)a_{ih}] b_h(\sigma_1) = 9.6 \times 10^{-3} = 9.6E^{-3}
\]

\[
\sigma_0 = \text{ship} \quad \sigma_1 = \text{frock} \quad \sigma_2 = \text{tops}
\]

Observations, \(\sigma_t\)
Step 2: Induction of Viterbi

\[
\begin{align*}
\sigma_0 &= \text{ship} \\
\sigma_1 &= \text{frock} \\
\sigma_2 &= \text{tops}
\end{align*}
\]

Observations, \( \sigma_t \)

\[
\begin{align*}
\delta_s(0) &= 0, a_{ss} = 1.0, \quad \therefore \delta_s(0)a_{ss} = 0 \\
\delta_h(0) &= 0.06, a_{hs} = 0.2, \quad \therefore \delta_h(0)a_{hs} = 0.012 \\
\delta_d(0) &= 0.08, a_{ds} = 0.1, \quad \therefore \delta_d(0)a_{ds} = 0.008
\end{align*}
\]
Step 2: Induction of Viterbi

\[ \delta_h(0)a_{hh} = 0.012, \quad b_s(frock) = 0.3 \]

\[ \therefore \max_i [\delta_i(0)a_{is}] b_s(\sigma_1) = 3.6 \times 10^{-3} = 3.6E^{-3} \]

**Observations, \( \sigma_t \)**

- \( \sigma_0 = ship \)
- \( \sigma_1 = rock \)
- \( \sigma_2 = tops \)
Marriage

- Jack is looking at Anne. Anne is looking at George. Jack is married but George is not.

Is a married person looking at an unmarried person?

A. Yes
B. No
C. Not enough information
Answer: Marriage

- Jack is looking at Anne. Anne is looking at George. Jack is married but George is not.

Is a married person looking at an unmarried person? **YES.**

Moral: something about hidden states, I guess
Step 2: Induction of Viterbi

\[ \begin{align*}
\sigma_0 &= \text{ship} \\
\sigma_1 &= \text{frock} \\
\sigma_2 &= \text{tops}
\end{align*} \]

For observations, \( \sigma_t \):

- \( \delta_s(2) = \max_i [\delta_i(1)a_{is}] b_s(\sigma_2) \)
- \( \psi_s(2) = \arg\max_i [\delta_i(1)a_{is}] \)

- \( \delta_h(2) = \max_i [\delta_i(1)a_{ih}] b_h(\sigma_2) \)
- \( \psi_h(2) = \arg\max_i [\delta_i(1)a_{ih}] \)

- \( \delta_d(2) = \max_i [\delta_i(1)a_{id}] b_d(\sigma_2) \)
- \( \psi_d(2) = \arg\max_i [\delta_i(1)a_{id}] \)

Values:

- \( 0 \rightarrow 3.6E^{-3} \rightarrow h \)
- \( 0.06 \rightarrow 9.6E^{-3} \rightarrow h \)
- \( 0.08 \rightarrow 1.92E^{-2} \rightarrow d \)
Step 2: Induction of Viterbi

\[ \delta_s(1) = 3.6E^{-3}, a_{sd} = 0, \quad \therefore \delta_s(1)a_{sd} = 0 \]

\[ \delta_h(1) = 9.6E^{-3}, a_{hd} = 0, \quad \therefore \delta_h(1)a_{hd} = 0 \]

\[ \delta_d(1) = 1.92E^{-2}, a_{dd} = 0.4, \quad \therefore \delta_d(1)a_{dd} = 0.00768 \]

\[ \psi_h(2) = \arg\max_i [\delta_i(1)a_ih] \]

\[ \delta_d(2) = \max_i [\delta_i(1)a_is] b_s(\sigma_2) \]

\[ \psi_d(2) = \arg\max_i [\delta_i(1)a_id] \]

\[ \sigma_0 = \text{ship} \quad \sigma_1 = \text{frock} \quad \sigma_2 = \text{tops} \]

Observations, \( \sigma_t \)
Step 2: Induction of Viterbi

Continuing...

<table>
<thead>
<tr>
<th>Observation, $\sigma_t$</th>
<th>$\delta_s(2) = 1.92E^{-3} \cdot 0.01$</th>
<th>$\delta_h(2) = 9.6E^{-3} \cdot 0.4$</th>
<th>$\delta_d(2) = 7.68E^{-3} \cdot 0.05$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma_0 = ship$</td>
<td>$h$</td>
<td>$h$</td>
<td>$d$</td>
</tr>
<tr>
<td>$\sigma_1 = f\text{rock}$</td>
<td>$3.6E^{-3}$</td>
<td>$9.6E^{-3}$</td>
<td>$1.92E^{-2}$</td>
</tr>
<tr>
<td>$\sigma_2 = tops$</td>
<td>$0$</td>
<td>$0$</td>
<td>$0$</td>
</tr>
</tbody>
</table>

$\sigma_0 = ship$, $\sigma_1 = f\text{rock}$, $\sigma_2 = tops$
Step 2: Induction of Viterbi

Note:
When computing $\delta_s(2)$, you will have a tie between $\delta_h(1)a_{hs}$ and $\delta_d(1)a_{ds}$.

It does not (yet) matter how you break this tie.

Observations, $\sigma_t$

$\sigma_0 = ship$
$\sigma_1 = f rock$
$\sigma_2 = tops$

$\delta_s(2) = 1.92E^{-3} \cdot 0.01$
$\psi_s(2) = h$

$\delta_h(2) = 9.6E^{-3} \cdot 0.4$
$\psi_h(2) = d$

$\delta_d(2) = 7.68E^{-3} \cdot 0.05$
$\psi_d(2) = d$
Step 3: Conclusion of Viterbi

Choose the best final state:

\[ Q_T^* = \arg\max_i \delta_i(T) \]

\( Q_T^* = h \)

Observations, \( \sigma_t \)

\( \sigma_0 = ship \quad \sigma_1 = frock \quad \sigma_2 = tops \)
Step 3: Conclusion of Viterbi

Recursively choose the best previous state:

\[ Q_{t-1}^* = \psi_{Q_t^*}(t) \]

Observations, \( \sigma_t \):
- \( \sigma_0 = \text{ship} \)
- \( \sigma_1 = \text{frock} \)
- \( \sigma_2 = \text{tops} \)
Step 3: Conclusion of Viterbi

Sequence probability:

\[ P(\mathcal{O}, Q^*; \theta) = \max_i \delta_i(T) \]

Observations, \( \sigma_t \):

- \( \sigma_0 = \text{ship} \)
- \( \sigma_1 = \text{frock} \)
- \( \sigma_2 = \text{tops} \)
Why did we choose $Q^* = \{q_0 \ldots q_T\}$?

- Recall the purpose of HMMs:
  - To represent multivariate systems where some variable is unknown/hidden/latent.

- Finding the best hidden-state sequence $Q^*$ allows us to:
  - Identify **unseen parts-of-speech** given words,
  - Identify **equivalent English** words given French words,
  - Identify **unknown phonemes** given speech sounds,
  - Decipher **hidden messages** from encrypted symbols,
  - Identify **hidden relationships** from gene sequences,
  - Identify **hidden market conditions** given stock prices,
  - ...

Working in the log domain

- Our formulation was

\[ Q^* = \arg\max_Q P(\mathcal{O}, Q; \theta) \]

this is equivalent to

\[ Q^* = \arg\min_Q -\log_2 P(\mathcal{O}, Q; \theta) \]

where

\[ -\log_2 P(\mathcal{O}, Q; \theta) = -\log_2 \left( \pi_{q_0} b_{q_0}(\sigma_0) \right) - \sum_{t=1}^{T} \log_2 \left( a_{q_{t-1}q_t} b_{q_t}(\sigma_t) \right) \]
Reading

• Manning & Schütze: Section 9.2—9.4.1


• Hidden Markov Model Toolkit (http://htk.eng.cam.ac.uk/) (if you’re interested)