COMP30027 Machine Learning
Sequential Classification

Semester 1, 2018

Jeremy Nicholson & Tim Baldwin & Karin Verspoor

Lecture Outline

Structured Classification

- To date, we have always considered each instance independently, but in many tasks, there is "structure" between instances, e.g.:
  - sequential structure (e.g. time series analysis, speech recognition, genomic data)
  - hierarchical structure (e.g. classifying web pages within a web site)
  - graph structure (e.g. deriving an "influence matrix" for a social network)
- This calls for **structured classification** models which are able to capture the interaction between instances

Source(s): Blunsom [2007]

Markov Chains

- A Markov chain is a finite state automaton (FSA) of the form $\mu = (A, \Pi)$ over a set $S = \{s_i\}$ of states, where:

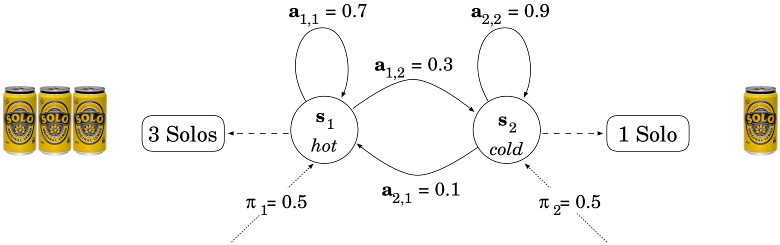
  $A = \{a_{ij}\}$ transition probability matrix; $\forall i : \sum_j a_{ij} = 1$

  $\Pi = \{\pi_i\}$ the initial state distribution; $\sum_i \pi_i = 1$

- Markov chains encode the assumption that a state q_i only depends on the immediately preceding state:

$$P(q_i | q_1 \ldots q_{i-1}) = P(q_i | q_{i-1})$$

Example Markov Chain: Solo Man



Example Calculation based on Solo Man

- What is the probability of observing 3-Solos, 3-Solos, 1-Solo?

Example Calculation based on Solo Man

- What is the probability of observing 3-Solos, 3-Solos, 1-Solo?

$$
\begin{aligned}
P(3,3,1) &= 0.5 \times 0.7 \times 0.3 \\
&= 0.105
\end{aligned}
$$

Lecture Outline

Hidden Markov Models

- But what if there are different possibilities attached to each observation, rather than a unique observation per state?

  $\Rightarrow$ *we see "observations", but we want to know "hidden states"*

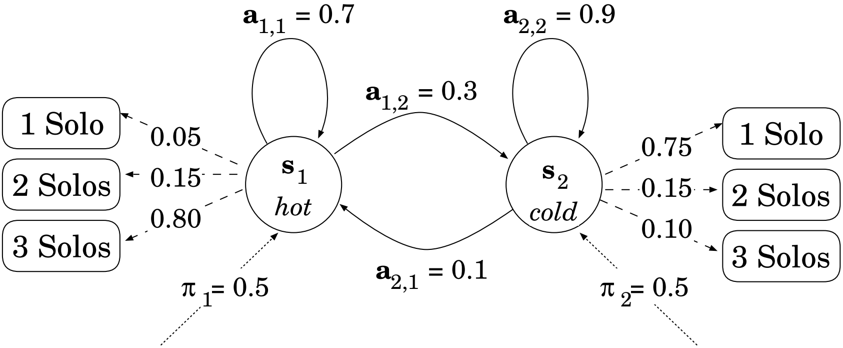
- Hidden Markov models (HMMs) take the form $\mu = (A, B, \Pi)$ over S and $O = \{o_k\}$ observations:

$A = \{a_{ij}\}$	transition probability matrix; $\forall i : \sum_j a_{ij} = 1$
$B = \{b_i(o_k)\}$	output probability matrix; $\forall i : \sum_k b_i(o_k) = 1$
$\Pi = \{\pi_i\}$	the initial state distribution; $\sum_i \pi_i = 1$

- HMMs make the additional independence assumption:

$$P(o_i | q_1, ..., q_i, o_1, ..., o_{i-1}) = P(o_i | q_i)$$

Example HMM: Solo Man with Something to Hide



Fundamental Problems Associated with HMM

- **Evaluation**: Given an HMM μ and observation sequence Ω, determine the likelihood $P(\Omega|\mu)$
- **Decoding**: Given an HMM μ and observation sequence Ω, determine the most probable hidden state sequence Q
- **Learning**: Given an observation sequence Ω and the set of possible states S and observations O in an HMM, learn the HMM parameters A, B and Π

Source(s): Rabiner [1989]

Evaluation based on Solo Man with Something to Hide

- What is the probability of observing 3-Solos, 3-Solos, 1-Solo?

  Easy to calculate if we know that the associated days were hot, hot, cold ... ($\mathcal{O}(T)$)

  Harder to calculate if we don't know the "hidden state" sequence ... ($\mathcal{O}(TN^T)$)

($T = |\Omega|$ and $N = |S|$)

Evaluation

- Probability of the state sequence Q:

$$P(Q|\mu) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}$$

- Probability of observation sequence Ω for state sequence Q:

$$P(\Omega|Q,\mu) = \prod_{t=1}^{T} P(o_t|q_t,\mu)$$

- Probability of a given observation sequence Ω:

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

Source(s): Rabiner [1989]

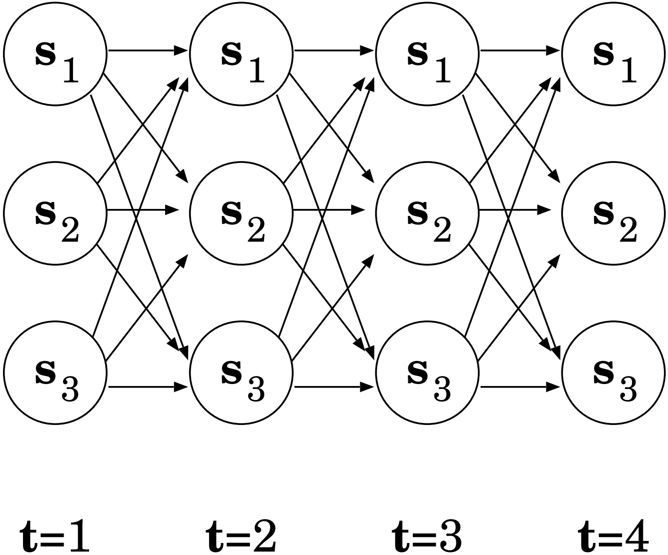
The Forward Algorithm

- Efficient computation of total probability (i.e. $P(\Omega|\mu)$) through "dynamic programming"
- Probability of the first t observations is the same for all possible $t+1$ length sequences
- Define forward probability:

$$\alpha_t(i) = P(o_1 o_2 \ldots o_t, q_t = s_i|\mu)$$

  i.e., the probability of the partial observation sequence, $o_1 o_2 \ldots o_t$, and state s_i at time t, given the model μ
- By caching forward probabilities in a trellis we can avoid redundant calculations
- The Backward Algorithm is just the reverse, i.e. start at T and work backwards through the trellis

The Forward Algorithm: Trellis



The Forward Algorithm

- Initialisation:

$$\alpha_1(i) = \pi_i b_i(o_1), \;\; i \in [i, N]$$

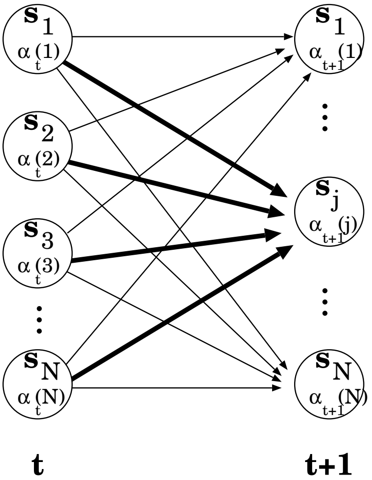
- Induction:

$$\alpha_{t+1}(i) = \left(\sum_{j=1}^{N} \alpha_t(j) a_{ji} \right) b_i(o_{t+1}), \;\; t \in [1, T-1], \;\; i \in [1, N]$$

- Termination:

$$P(\Omega|\mu) = \sum_{i=1}^{N} \alpha_T(i)$$

The Forward Algorithm: Trellis Traversal



Returning to our Example ...

- Initialisation/induction:

	$t=1$	$t=2$	$t=3$
$\alpha_t(hot)$:	0.5×0.8 $= 0.4$	$[0.4 \times 0.7$ $+0.05 \times 0.1] \times 0.8$ $= 0.228$	$[0.228 \times 0.7$ $+0.0165 \times 0.1] \times 0.05$ $= 0.0080625$
$\alpha_t(cold)$:	0.5×0.1 $= 0.05$	$[0.4 \times 0.3$ $+0.05 \times 0.9] \times 0.1$ $= 0.0165$	$[0.228 \times 0.3$ $+0.0165 \times 0.9] \times 0.75$ $= 0.0624375$

- Termination:

$$\begin{aligned} P(\text{3-Solos}, \text{3-Solos}, \text{1-Solo}|\mu) &= 0.0080625 + 0.0624375 \\ &= 0.0705 \end{aligned}$$

Decoding based on Solo Man with Something to Hide

- Given the observation 3-Solos, 3-Solos, 1-Solo, what is the most probable weather sequence?

Decoding based on Solo Man with Something to Hide

- Given the observation 3-Solos, 3-Solos, 1-Solo, what is the most probable weather sequence?

  Could enumerate all the hidden state sequences brute-force and sort ... ($\mathcal{O}(TN^T + N^T \log N^T)$)

  The Viterbi algorithm gives us a much more efficient method

Viterbi Algorithm: Preliminaries

- Introduce notation for the maximum probability for a partial sequence along a single path:

$$\delta_t(i) = \max_{q_1 q_2 .. q_{t-1}} P(q_1 q_2 ... q_{t-1}, o_1 o_2 ... o_t, q_t = s_i | \mu)$$

Source(s): Rabiner [1989]

The Viterbi Algorithm I

- Initialisation:

$$
\begin{aligned}
\delta_1(i) &= \pi_i b_i(o_1),\;\; i \in [1, N] \\
\psi_1(i) &= 0
\end{aligned}
$$

- Induction:

$$
\begin{aligned}
\delta_t(i) &= \max_{j \in [1,N]} (\delta_{t-1}(j) a_{ji}) b_i(o_t),\;\; t \in [2, T],\;\; i \in [1, N] \\
\psi_t(i) &= \arg\max_{j \in [1,N]} (\delta_{t-1}(j) a_{ji}),\;\; t \in [2, T],\;\; i \in [1, N]
\end{aligned}
$$

The Viterbi Algorithm II

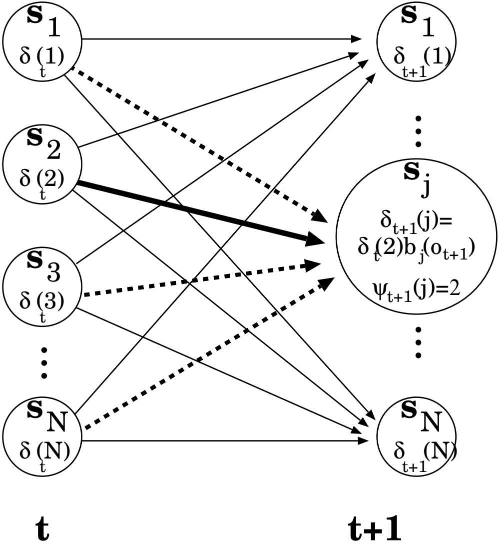
- Termination:

$$
\begin{aligned}
P_{\mathrm{best}} &= \max_{i \in [1,N]} \delta_T(i) \\
q_T &= \arg\max_{i \in [1,N]} \delta_T(i)
\end{aligned}
$$

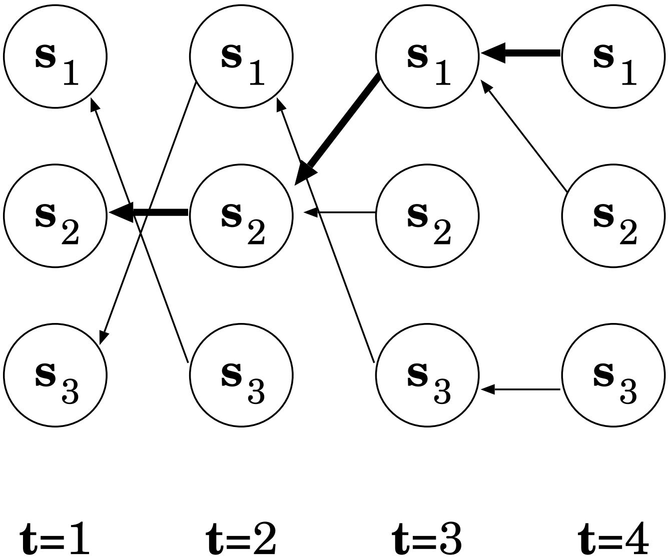
- Backtrack to establish the best path:

$$
q_t = \psi_{t+1}(q_{t+1}), \; t = T-1, T-2, \ldots, 1
$$

The Viterbi Algorithm: Induction



The Viterbi Algorithm: Backtrace



Returning again to our Example ... I

- Initialisation/induction:

	$t = 1$	$t = 2$	$t = 3$
$\delta_t(hot)$:	0.5×0.8 $= 0.4$	$\max(0.4 \times 0.7,$ $0.05 \times 0.1) \times 0.8$ $= 0.224$	$\max(0.224 \times 0.7,$ $0.012 \times 0.1) \times 0.05$ $= 0.00784$
$\psi_t(hot)$	0	$\leftarrow$ *hot*	$\leftarrow$ *hot*
$\delta_t(cold)$:	0.5×0.1 $= 0.05$	$\max(0.4 \times 0.3,$ $0.05 \times 0.9) \times 0.1$ $= 0.012$	$\max(0.224 \times 0.3,$ $0.012 \times 0.9) \times 0.75)$ $= 0.0504$
$\psi_t(cold)$	0	$\nwarrow$ *hot*	$\nwarrow$ *hot*

Observation sequence: 3-Solos, 3-Solos, 1-Solo

Returning again to our Example ... II

- Termination/backtracking:

$$
\begin{aligned}
P_{\text{best}} &= 0.0504 \\
q_T &= \textit{cold} \\
q_{T-1} &= \textit{hot} \\
q_{T-2} &= \textit{hot}
\end{aligned}
$$

$\rightarrow$ *the most probable sequence of hidden states which produces the observation sequence 3-Solos, 3-Solos, 1-Solo is hot, hot, cold*

Learning HMMs: The Supervised Case

- Assume we have labelled data, it is possible to use simple MLE to learn the parameters of our model:

$$P(q_j|q_i) = \frac{freq(q_i, q_j)}{freq(q_i)} = a_{ij}$$

$$P(o_k|q_i) = \frac{freq(o_k, q_i)}{freq(q_i)} = b_i(o_k)$$

$$P(q_i|\mathrm{START}) = \frac{freq(\mathrm{START}, q_i)}{\sum_j freq(\mathrm{START}, q_j)} = \pi_i$$

- Can also train models in an unsupervised fashion using Baum-Welch algorithm (EM)

HMMs: Reflections

- Highly efficient approach to structured classification, but limited representation of context (sequence of 2 only)
- As with NB, HMM tends to suffer from floating point underflow
    - use logs for Viterbi Algorithm
    - use scaling coefficients for Forward Algorithm
- As with most generative models, it's hard to add ad hoc features

Lecture Outline

Other Structured Classifiers

- **Maximum Entropy Markov Models:** logistic regression (= "maximum entropy") model where we also condition on (properties of) the observation:

  $$\hat{c} = \arg\max_{T} \prod_{i} P(q_i | o_i, q_{i-1})$$

  Unlike HMMs, it's possible to add extra features indiscriminately *as well as* capturing the (unidirectional) tag interactions
- **Conditional Random Fields:** extension of logistic regression where we optimise over the full tag sequence

Source(s): Blunsom [2007], Lafferty et al. [2001]

Lecture Outline

Summary

- What is structured classification?
- How do we evaluate a HMM?
- How do we decode a HMM?
- How do you train an HMM given labelled training data?
- What are limitations of HMMs, and what more sophisticated sequential classification algorithms are there?

References I

Philip Blunsom. *Structured Classification for Multilingual Natural Language Processing*. PhD thesis, University of Melbourne, 2007.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289, Williamstown, USA, 2001.

Lawrence R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), 1989.