CS 188: Artificial Intelligence  
Fall 2010

Lecture 21: Speech / ML  
11/9/2010

Dan Klein – UC Berkeley

---

Announcements

- **Assignments:**
  - Project 2: In glookup
  - Project 4: Due 11/17
  - Written 3: Out later this week

- Contest out now!

- Reminder: surveys (results next lecture)
Contest!

Today

- HMMs: Most likely explanation queries
- Speech recognition
  - A massive HMM!
  - Details of this section not required
- Start machine learning
Speech and Language

- **Speech technologies**
  - Automatic speech recognition (ASR)
  - Text-to-speech synthesis (TTS)
  - Dialog systems

- **Language processing technologies**
  - Machine translation
  - Information extraction
  - Web search, question answering
  - Text classification, spam filtering, etc...

---

**HMMs: MLE Queries**

- **HMMs defined by**
  - States $X$
  - Observations $E$
  - Initial distr: $P(X_1)$
  - Transitions: $P(X|X_{-1})$
  - Emissions: $P(E|X)$

- **Query: most likely explanation:**
  \[ \arg \max_{x_{1:t}} P(x_{1:t}|e_{1:t}) \]
State Path Trellis

- State trellis: graph of states and transitions over time
- Each arc represents some transition $x_{t-1} \rightarrow x_t$
- Each arc has weight $P(x_t|x_{t-1})P(e_t|x_t)$
- Each path is a sequence of states
- The product of weights on a path is the seq’s probability
- Can think of the Forward (and now Viterbi) algorithms as computing sums of all paths (best paths) in this graph

Viterbi Algorithm

$$x_{1:T}^* = \arg \max_{x_{1:T}} P(x_{1:T}|e_{1:T}) = \arg \max_{x_{1:T}} P(x_{1:T}, e_{1:T})$$

$$m_t[x_t] = \max_{x_{1:t-1}} P(x_{1:t-1}, x_t, e_{1:t})$$

$$= \max_{x_{1:t-1}} P(x_{1:t-1}, e_{1:t-1})P(x_t|x_{t-1})P(e_t|x_t)$$

$$= P(e_t|x_t) \max_{x_{t-1}} P(x_t|x_{t-1}) \max_{x_{1:t-2}} P(x_{1:t-1}, e_{1:t-1})$$

$$= P(e_t|x_t) \max_{x_{t-1}} P(x_t|x_{t-1}) m_{t-1}[x_{t-1}]$$
Digitizing Speech

Speech in an Hour

- Speech input is an acoustic wave form

Graphs from Simon Arnfield’s web tutorial on speech, Sheffield:
http://www.psyc.leeds.ac.uk/research/cogn/speech/tutorial/
Frequency gives pitch; amplitude gives volume
- sampling at ~8 kHz phone, ~16 kHz mic (kHz=1000 cycles/sec)

Fourier transform of wave displayed as a spectrogram
- darkness indicates energy at each frequency

Part of [ae] from “lab”
- Complex wave repeating nine times
  - Plus smaller wave that repeats 4x for every large cycle
  - Large wave: freq of 250 Hz (9 times in 0.036 seconds)
  - Small wave roughly 4 times this, or roughly 1000 Hz
Acoustic Feature Sequence

- Time slices are translated into acoustic feature vectors (~39 real numbers per slice)
- These are the observations, now we need the hidden states \( X \)

State Space

- \( P(E|X) \) encodes which acoustic vectors are appropriate for each phoneme (each kind of sound)
- \( P(X|X') \) encodes how sounds can be strung together
- We will have one state for each sound in each word
- From some state \( x \), can only:
  - Stay in the same state (e.g. speaking slowly)
  - Move to the next position in the word
  - At the end of the word, move to the start of the next word
- We build a little state graph for each word and chain them together to form our state space \( X \)
HMMs for Speech

Transitions with Bigrams

Figure from Huang et al page 618
Decoding

- While there are some practical issues, finding the words given the acoustics is an HMM inference problem.
- We want to know which state sequence $x_{1:T}$ is most likely given the evidence $e_{1:T}$:
  \[
  x_{1:T}^* = \arg\max_{x_{1:T}} P(x_{1:T} | e_{1:T})
  \]
  \[
  = \arg\max_{x_{1:T}} P(x_{1:T}, e_{1:T})
  \]
- From the sequence $x$, we can simply read off the words.

End of Part II!

- Now we’re done with our unit on probabilistic reasoning.
- Last part of class: machine learning.
Machine Learning

- Up until now: how to reason in a model and how to make optimal decisions

- Machine learning: how to acquire a model on the basis of data / experience
  - Learning parameters (e.g. probabilities)
  - Learning structure (e.g. BN graphs)
  - Learning hidden concepts (e.g. clustering)

Parameter Estimation

- Estimating the distribution of a random variable
- **Elicitation:** ask a human (why is this hard?)
- **Empirically:** use training data (learning!)
  - E.g.: for each outcome \( x \), look at the *empirical rate* of that value:

\[
P_{\text{ML}}(x) = \frac{\text{count}(x)}{\text{total samples}}
\]

\[
P_{\text{ML}}(r) = \frac{1}{3}
\]

- This is the estimate that maximizes the *likelihood of the data*

\[
L(x, \theta) = \prod_i P_\theta(x_i)
\]
Estimation: Smoothing

- Relative frequencies are the maximum likelihood estimates

\[ \theta_{ML} = \arg \max_{\theta} P(X|\theta) \]
\[ = \arg \max_{\theta} \prod_{i} P_{\theta}(X_i) \quad \Rightarrow \quad P_{ML}(x) = \frac{\text{count}(x)}{\text{total samples}} \]

- In Bayesian statistics, we think of the parameters as just another random variable, with its own distribution

\[ \theta_{MAP} = \arg \max_{\theta} P(\theta|X) \]
\[ = \arg \max_{\theta} P(X|\theta)P(\theta)/P(X) \quad \Rightarrow \quad ??? \]
\[ = \arg \max_{\theta} P(X|\theta)P(\theta) \]

Estimation: Laplace Smoothing

- Laplace’s estimate:
  - Pretend you saw every outcome once more than you actually did

\[ P_{LAP}(x) = \frac{c(x) + 1}{\sum_x [c(x) + 1]} \]
\[ = \frac{c(x) + 1}{N + |X|} \]

- Can derive this as a MAP estimate with Dirichlet priors (see cs281a)
Estimation: Laplace Smoothing

- **Laplace’s estimate (extended):**
  - Pretend you saw every outcome \( k \) extra times
  \[
P_{LAP,k}(x) = \frac{c(x) + k}{N + k|X|}
\]
- What’s Laplace with \( k = 0 \)?
- \( k \) is the strength of the prior

- **Laplace for conditionals:**
  - Smooth each condition independently:
  \[
P_{LAP,k}(x|y) = \frac{c(x, y) + k}{c(y) + k|X|}
\]

\[
P_{LAP,0}(X) =
P_{LAP,1}(X) =
P_{LAP,100}(X) =
\]