Hidden Markov Models

Algorithms for NLP

September 25, 2014
Quick Review

• WFSAs
  – Deterministic (for scoring strings)

• Best path algorithm for WFSAs
General Case: Ambiguous WFSAs
Ambigious WFSAs

- A string may have more than one state sequence.
- Useful for representing ... ambiguity!
  - Weights define relative goodness of different paths.
- Recall from last time: the path can be used to encode an analysis.
  - E.g., if symbols are words, the state before or after might correspond to the word's part of speech.
Toward an Algorithm

• We can use weights to encourage or discourage paths.
• In an ambiguous FSA, there might be multiple paths for the same string.
• Important algorithm: find the best path for a string.
• Related algorithm: find the sum of all paths
A Tempting, Incorrect Algorithm
Tempting but Incorrect

- Pick the “best” start state.

\[ q_0 := \arg \max_{q \in I} \lambda(q) \]

- For \( t = 1 \) to \( |s| \):

\[ q_t := \arg \max_{q \in Q} w(q_{t-1}, s_t, q) \]
Why It’s Wrong In General

daz
Problems

• What if there is no transition out of $q_{t-1}$ that matches $s_t$?
• If there are any epsilons in the sequence, we *can't see them*!
  – This algorithm will never take an $\varepsilon$-edge.
• Something fishy about leaving out the stopping weights $\rho$!
• In general: a seemingly good decision early on could turn out to be very bad later on, since rewards and punishments can always be delayed until the very end.
ε-Transitions

• Let’s continue to ignore them
  – There is a weighted “remove epsilons” operation that preserves best paths.
An Easier Problem

• If I knew the best path \( \hat{e}_1 \cdots \hat{e}_{\ell-1} \) for the first \( \ell-1 \) symbols \( s_1 \ldots s_{\ell-1} \), then the decision for the final part would be easy:

\[
q_\ell := \arg \max_{q \in F} w(n(\hat{e}_{\ell-1}), s_\ell, q) \times \rho(q)
\]

• Note that this choice only depends on \( q_{\ell-1} \), not the whole path prefix \( q_1 \ldots q_{\ell-1} \).

• Manageable: best path prefix for \( s_1 \ldots s_{\ell-1} \) ending in each \( q \).
Toward a Recurrence

- Let \( \alpha(q, i) \) be the score of the best path for the length \( t \) prefix \( (s_1...s_t) \) ending in state \( q \).

\[
\alpha(q, t) = \max_{e_1 e_2 ... e_t : n(e_t) = q} \lambda(p(e_1)) \times \prod_{i=1}^{t} w(e_i) \quad \forall q \in Q, \forall t \in [1, \ell)
\]

\[
= \max_{e_1 e_2 ... e_t : n(e_t) = q} \lambda(p(e_1)) \times \left( \prod_{i=1}^{t-1} w(e_i) \right) \times w(\langle n(e_{t-1}), s_t, q \rangle)
\]

\[
= \alpha(q_{t-1}, t-1) \quad = \alpha(q_{t-1}, t-1)
\]

\[
= \max_{q_{t-1} \in Q} \alpha(q_{t-1}, t - 1) \times w(\langle q_{t-1}, s_t, q \rangle)
\]
Recovering the Best Path

• We are given the prefix best-path scores:
  \[ \alpha : Q \times \{0, 1, ..., \ell\} \rightarrow \mathbb{R}_{\geq 0} \]

\[ q_\ell := \arg\max_q \alpha(q, \ell) \times \rho(q) \]

\textbf{for } t = \ell - 1 \textbf{ to } 0

\[ q_t := \arg\max_q \alpha(q, t) \times w(q, s_{t+1}, q_{t+1}) \]
$\alpha(q, i)$

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>...</th>
<th>$\ell - 2$</th>
<th>$\ell - 1$</th>
<th>$\ell$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_5$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_6$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\[ \forall q \in I, \quad \alpha(q, 0) \leftarrow \lambda(q) \]
\[ \alpha(q, i) \]

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>...</th>
<th>( \ell )-2</th>
<th>( \ell )-1</th>
<th>( \ell )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( q_0 )</td>
<td>( \lambda(q_0) )</td>
<td></td>
<td></td>
<td>[</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_1 )</td>
<td>( \lambda(q_1) )</td>
<td></td>
<td></td>
<td>[</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_2 )</td>
<td>( \lambda(q_2) )</td>
<td></td>
<td></td>
<td>[</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_3 )</td>
<td>( \lambda(q_3) )</td>
<td></td>
<td></td>
<td>[</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_4 )</td>
<td>( \lambda(q_4) )</td>
<td></td>
<td></td>
<td>[</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_5 )</td>
<td>( \lambda(q_5) )</td>
<td></td>
<td></td>
<td>[</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_6 )</td>
<td>( \lambda(q_6) )</td>
<td></td>
<td></td>
<td>[</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
\forall q \in Q, \quad \alpha(q, 1) = \max_{e \in (Q \times \{s_1\} \times \{q\} \cap E)} \alpha(p(e), 0) \times w(e)
\]
\[ \alpha(q, i) \]

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>...</th>
<th>(\ell)-2</th>
<th>(\ell)-1</th>
<th>(\ell)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(q_0)</td>
<td>(\lambda(q_0))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(q_1)</td>
<td>(\lambda(q_1))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(q_2)</td>
<td>(\lambda(q_2))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(q_3)</td>
<td>(\lambda(q_3))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(q_4)</td>
<td>(\lambda(q_4))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(q_5)</td>
<td>(\lambda(q_5))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(q_6)</td>
<td>(\lambda(q_6))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ \forall q \in Q, \quad \alpha(q, 2) = \max_{e \in (Q \times \{s_2\} \times \{q\} \cap E)} \alpha(p(e), 1) \times w(e) \]
\[ \alpha(q, i) \]

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>...</th>
<th>\ell-2</th>
<th>\ell-1</th>
<th>\ell</th>
</tr>
</thead>
<tbody>
<tr>
<td>( q_0 )</td>
<td>( \lambda(q_0) )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_1 )</td>
<td>( \lambda(q_1) )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_2 )</td>
<td>( \lambda(q_2) )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_3 )</td>
<td>( \lambda(q_3) )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_4 )</td>
<td>( \lambda(q_4) )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_5 )</td>
<td>( \lambda(q_5) )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_6 )</td>
<td>( \lambda(q_6) )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( \forall q \in Q, \forall i \in [1, \ell] \) \quad \alpha(q, t) = \max_{e \in (Q \times \{s_t\} \times \{q\} \cap E)} \alpha(p(e), t - 1) \times w(e)
Constructing Prefix Best-Path Scores

Input string is \((s_1 \ s_2 \ ... \ s_\ell)\)
Goal: construct \(\alpha : Q \times \{0, 1, ..., \ell\} \rightarrow \mathbb{R}_{\geq 0}\)

set all \(\alpha(q, i) := 0\)
for each \(q \in I\): \(\alpha(q, 0) := \lambda(q)\)
for \(t = 1\) to \(\ell\):
  for each \(q \in Q\):
    for each \(q' \in Q\):
      \[
      \alpha(q, t) := \max\{ \alpha(q, t), \alpha(q', t-1) \times w(q', s_t, q) \}
      \]
High-Level Best WFSA Path Algorithm

• First, construct the prefix best-path scores $\alpha$.
• Then recover the path.

• Alternative: maintain pointers to each “argmax” when constructing $\alpha$, then follow the pointers to recover the path.

• Runtime and space requirements for this algorithm, in $|Q|$ and in $\ell$?
Handling $\varepsilon$ Transitions

initialize all $\alpha(q, i)$ to 0
for each $q$: $\alpha(q, 0) := \pi(q)$
for $i = 1$ to $n$:
  for each $q$:
    for each $q'$:
      $\alpha(q, i) := \max\{ \alpha(q, i), \alpha(q', i-1) \times \delta(q', s_i, q) \}$
repeat until $\alpha(\ast, i)$ converge:
  for each $q$:
    for each $q'$:
      $\alpha(q, i) := \max\{ \alpha(q, i), \alpha(q', i) \times \delta(q', \varepsilon, q) \}$
Problems with $\varepsilon$

- Runtime is much harder to analyze, because of $\varepsilon$ cycles.
- Two cases:
  - Dampening cycles, where hypothesizing lots of $\varepsilon$ transitions makes the path's score worse. Example: $\delta(q, \varepsilon, q) < 1$
  - Amplifying cycles, e.g., $\delta(q, \varepsilon, q) > 1$. 
WFSA

weighted languages disambiguation

FSA

regular languages

FST

regular relations string-to-string mappings
WFSA
weighted languages
disambiguation

FSA
regular languages

WFST
weighted string-to-string mappings

FST
regular relations
string-to-string mappings
Beyond WFSAs: WFSTs

- Weighted finite-state *transducers*: combine the two-tape idea from last time with the weighted idea from today.
  - Very general framework; key operation is **weighted composition**.
  - Best-path algorithm variations: best path and output given input, best path given input and output, ...

- WFSAs then become just WFSTs encoding the identity relation
Weights Beyond Numbers

• Boolean weights = traditional FSAs/FSTs.
• Real weights = the case we explored so far.
• Many other semirings (sets of possible weight values and operations for combining weights)
• A semiring is
  – a set of values ("real numbers", "natural numbers", "strings", "points in R^2")
  – Addition +
  – Multiplication x
  – Zero
  – One
WFSAs and WFSTs: Algorithms

• OpenFST documentation contains high-level view of most algorithms you think you want, and pointers.

• The set of people developing general WFST algorithms largely overlaps with the OpenFST team.
  – Mehryar Mohri, Cyril Allauzen, Michael Reilly

• This is an active area of research!
From WFSAs to HMMs
## Weighted Finite-State Automaton

<table>
<thead>
<tr>
<th>Element</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q$</td>
<td>Finite set of states</td>
</tr>
<tr>
<td>$\Sigma$</td>
<td>Finite vocabulary</td>
</tr>
<tr>
<td>$I \subseteq Q$</td>
<td>Set of initial states</td>
</tr>
<tr>
<td>$F \subseteq Q$</td>
<td>Set of final states</td>
</tr>
<tr>
<td>$E \subseteq (Q \times (\Sigma \cup {\varepsilon}) \times Q)$</td>
<td>Set of transitions (edges)</td>
</tr>
<tr>
<td>$\lambda : I \rightarrow \mathbb{R}_{\geq 0}$</td>
<td>Initial weights</td>
</tr>
<tr>
<td>$\rho : F \rightarrow \mathbb{R}_{\geq 0}$</td>
<td>Final weights</td>
</tr>
<tr>
<td>$w : E \rightarrow \mathbb{R}_{\geq 0}$</td>
<td>Transition weights</td>
</tr>
</tbody>
</table>
First Change: Probability

• Probabilities are a special kind of weight.
• Informally, there are three rules:
  – All weights between zero and one.
  – At each “local position,” the weights of all competing possibilities have to sum to one.
  – Over all possible outcomes (beginning to end), the total sum of weights has to be one.
• The main operations when you deal in probabilities:
  – Multiplying together (conditional) probabilities of events
  – Adding probabilities to calculate marginals
  – Using data to estimate probabilities (e.g., counting)
## Probabilistic Finite-State Automaton

<table>
<thead>
<tr>
<th>Element</th>
<th>Definition</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q$</td>
<td>Finite set of states</td>
<td></td>
</tr>
<tr>
<td>$\Sigma$</td>
<td>Finite vocabulary</td>
<td></td>
</tr>
<tr>
<td>$I=Q$</td>
<td>Set of initial states</td>
<td></td>
</tr>
<tr>
<td>$F={q_f}$</td>
<td>Final state</td>
<td>$F \cap Q = {}$</td>
</tr>
<tr>
<td>$E=Q \times \Sigma \times (Q \cup F)$</td>
<td>Set of transitions</td>
<td></td>
</tr>
<tr>
<td>$\lambda : I \to [0,1]$</td>
<td>Initial weights</td>
<td>$\Sigma_q = 1$</td>
</tr>
<tr>
<td>$\rho : F \to {1}$</td>
<td>Final weights</td>
<td></td>
</tr>
<tr>
<td>$w : E \to [0,1]$</td>
<td>Transition weights</td>
<td>$\forall q, \Sigma_{r,s} w(q,s,r) = 1$</td>
</tr>
</tbody>
</table>
Differences?

- All paths have scores between 0 and 1.
- These conditions are sufficient to show that the sum of all start-to-finish paths is one.
  - Some of those paths might be infinitely long, though!
- No transitions out of $q_f$. 
Second Change: Factoring Transitions

• Define \( w : Q \times \Sigma \times Q \rightarrow [0, 1] \) as a product of two parts:

\[
  w(q, s, q') = \eta(s \mid q) \times \gamma(q' \mid q)
\]

• Instead of transitioning in one move, the model first emits a symbol (\( \eta \)) then transitions (\( \gamma \)).

• A PFSA with this property is called an HMM.
Effects?

• It's easy to see that every HMM is a PFSA.
• Can we go in the other direction?
  – Not in general.
“Silent States”

• Some definitions of HMMs allow states to be silent (always, or with some probability).
  – Related to $\varepsilon$ transitions.

• In NLP, we do not usually use silent states, so I will remain silent on the matter.
  – This simplifies things.
The Conventional View of HMMs
Probabilistic Generative Stories

• This view focuses on the HMM as a way of generating strings.

• Probabilistic generative models are a huge class of techniques.
  – In machine learning: naïve Bayes, mixtures of Gaussians, latent Dirichlet allocation (LDA), ... and many more; emphasis there is on recovering the parameters of a model from the data.
The $n$-Gram Story

• Fix each of $s_{-n+2}$, $s_{-n+3}$, ..., $s_0$ to the START symbol.
• For $i = 1$, 2, ...:
  - Pick $s_i$ according to the distribution $p(\cdot \mid s_{i-n+1}, s_{i-n+2}, \ldots, s_{i-1})$
  - Exit loop if $s_i$ is the STOP symbol.

“The Markov assumption”: the probability of a word given its history depends only on its $(n – 1)$ most recent predecessors.
The HMM Story

• Pick a start state $q$ according to $\pi(q)$.
• If state $q$ is the stop state, then stop; otherwise:
  – Emit symbol $s$ with probability $\eta(s \mid q)$.
  – Transition to $q'$ with probability $\gamma(q' \mid q)$.

“The Markov assumption”: the probability of a word given its history only depends on the current state.
The HMM Story, in Math

\[ p(q_0, s_1, q_1, \ldots, s_n, q_n) = \lambda(q_0) \times \left( \prod_{i=0}^{n-1} \eta(s_{i+1} | q_i) \times \gamma(q_{i+1} | q_i) \right) \times \rho(q_n) \]
The Two Views, Again

• Declarative: HMMs (like all WFSAs) assign scores to paths by multiplying together local weights.
• Procedural: HMMs imply an algorithm for generating data.
  – In general, WFSAs do not. (Why?)
  – What we really want is an algorithm for recovering the states, given the symbols.
C53
one symbol die per state:
one “next state” die per state:
one “stop or continue” coin per state:
C53  C23

I want

one symbol die per state:

...
one “next state” die per state:
I want one “stop or continue” coin per state:
one symbol die per state:

C53  C23  C2

I want a
I want a one “next state” die per state:
C53  C23  C2  C5

I want a

one “stop or continue” coin per state:
C53  C23  C2  C5

I want a flight

one symbol die per state:
Note: N-Gram Models vs. HMMs

• N-gram models are a special case of “Markov models,” in which the state is fully determined by the words.

• “Markov,” as opposed to “hidden Markov,” because the states are known (not hidden) given the sequence of symbols (recall, states = histories).

  Markov models ⊂ deterministic WFSAs
  HMMs ⊂ ambiguous WFSAs
Where Do Weights Come From?

• In the general case of WFSAs, it is not clear where weights should come from.
  – There are lots of ways you could imagine doing it.

• In the HMM case, the probabilistic view suggests that we use data and statistics.
  – General framework for extracting probabilistic model parameters from data: maximum likelihood estimation.
  – Given symbols and their states, we can use relative frequency estimates.
Relative Frequency Estimation

• “Count and normalize.”
• For conditional probabilities, the ratio of the number of times the event *did* happen to the number of times it *might have* happened.

\[
\hat{X}(q) = \frac{\text{count}(q_{\text{initial}} = q)}{\#\text{sequences}}
\]

\[
\eta(s \mid q) = \frac{\text{count}(q \text{ emits } s)}{\text{count}(q_{\text{nonfinal}} = q)}
\]

\[
\gamma(q' \mid q) = \frac{\text{count}(q \text{ transitions to } q')}{\text{count}(q_{\text{nonfinal}} = q)}
\]

\[
\rho(q) = \frac{\text{count}(q_{\text{final}} = q)}{\text{count}(q)}
\]
The Viterbi Algorithm
Most Probable State Sequence

• For WFSAs, we talked about the “best” path.
• Probabilistic models allow us to say “most likely” or “most probable” path.

\[ \arg \max_{q_0, \ldots, q_n} p(q_0, s_1, q_1, \ldots, s_n, q_n) \]

\[ = \arg \max_{q_0, \ldots, q_n} \pi(q_0) \times \left( \prod_{i=0}^{n-1} \eta(s_{i+1} \mid q_i) \times \gamma(q_{i+1} \mid q_i) \right) \times \xi(q_n) \]

– For a PFSA, the algorithm is identical to what it was before.
– HMMs factor \( \delta \) into two parts (\( \eta \) and \( \gamma \)); this changes the algorithm a little bit.
Conditional vs. Joint

- HMMs define the **joint**: $p(\text{states, symbols})$
- We want the **conditional**: $p(\text{states} \mid \text{symbols})$
- It turns out that:
  \[
  \arg \max_q p(q \mid s) \\
  = \arg \max_q p(q, s) / p(s) \\
  = \arg \max_q p(q, s)
  \]
- So we can focus on maximizing the **joint**.
Example: POS Tagging

I suspect the present forecast is pessimistic.

CD JJ DT JJ NN NNS JJ .
NN NN JJ NN VB VBZ
NNP VB NN RB VBD
PRP VBP NNP VB VBN
  VBP VBP VBP

4 4 5 5 5 2 1 1

4,000 possible state sequences!
Naïve Solutions

• List all the possibilities in $Q^n$.
  – Correct.
  – Inefficient.

• Work left to right and greedily pick the best $q_i$ at each point, based on $q_{i-1}$ and $s_{i+1}$.
  – Not correct; solution may not be the most probable path.
Interactions

• Each word's label depends on the word, and nearby labels.
• But given adjacent labels, others do not matter.

I suspect the present forecast is pessimistic.

CD JJ DT JJ NN NNS JJ .
NN NN JJ NN VB VBZ
NNP VB NN RB VBD
PRP VBP NNP VB VBN
VBP VBP VBP

(arrows show most preferred label by each neighbor)
High-Level Viterbi Algorithm

• First, construct the prefix best-path scores $\alpha$, with back pointers.
• Then recover the path.
Constructing Prefix Best-Path Scores

- Input string is \((s_1 s_2 \ldots s_n)\)
- Goal: Construct \(\alpha : Q \times \{0, 1, \ldots, n\} \rightarrow \mathbb{R}_{\geq 0}\)

initialize all \(\alpha(q, i)\) to 0
for each \(q\): \(\alpha(q, 0) := \pi(q)\)
for \(i = 1\) to \(n\):
  for each \(q\):
    for each \(q'\):
      \(\alpha(q, i) := \max\{ \alpha(q, i), \alpha(q', i-1) \times \eta(s_i | q') \times \gamma(q | q') \} \)
Constructing Prefix Best-Path Scores

• Input string is \((s_1 s_2 \ldots s_n)\)
• Goal: construct \(\alpha : Q \times \{0, 1, \ldots, n\} \rightarrow \mathbb{R}_{\geq 0}\)

initialize all \(\alpha(q, i)\) to 0
for each \(q\): \(\alpha(q, 0) := \pi(q)\)
for \(i = 1\) to \(n\):
  for each \(q\):
    for each \(q'\):
      \(\alpha(q, i) := \max\{ \alpha(q, i), \alpha(q', i-1) \times \delta(q', s_i, q) \} \)
I suspect the present forecast is pessimistic.

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th>suspect</th>
<th>the</th>
<th>present</th>
<th>forecast</th>
<th>is</th>
<th>pessimistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD</td>
<td>3E-7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DT</td>
<td></td>
<td>3E-8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JJ</td>
<td>1E-9</td>
<td>1E-12</td>
<td>3E-12</td>
<td></td>
<td></td>
<td></td>
<td>7E-23</td>
</tr>
<tr>
<td>NN</td>
<td>4E-6</td>
<td>2E-10</td>
<td>1E-13</td>
<td>6E-13</td>
<td>4E-16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NNP</td>
<td>1E-5</td>
<td>4E-13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NNS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1E-21</td>
</tr>
<tr>
<td>PRP</td>
<td>4E-3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2E-14</td>
</tr>
<tr>
<td>VB</td>
<td>6E-9</td>
<td>3E-15</td>
<td>2E-19</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6E-18</td>
</tr>
<tr>
<td>VBN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4E-18</td>
</tr>
<tr>
<td>VBP</td>
<td>5E-7</td>
<td>4E-14</td>
<td>4E-15</td>
<td>9E-19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6E-18</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2E-24</td>
</tr>
</tbody>
</table>
I suspect the present forecast is pessimistic.
I suspect the present forecast is pessimistic.
Another Example

<table>
<thead>
<tr>
<th></th>
<th>time</th>
<th>flies</th>
<th>like</th>
<th>an</th>
<th>arrow</th>
<th>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td></td>
<td></td>
<td></td>
<td>10e-15</td>
<td>6e-21</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td></td>
<td>8e-13</td>
<td></td>
<td>1e-19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jj</td>
<td></td>
<td>6e-14</td>
<td></td>
<td>2e-16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>nn</td>
<td>2e-4</td>
<td></td>
<td></td>
<td>3e-16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>nnp</td>
<td></td>
<td></td>
<td></td>
<td>1e-16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vb</td>
<td>2e-7</td>
<td>1e-14</td>
<td></td>
<td>1e-19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vbp</td>
<td></td>
<td>8e-16</td>
<td></td>
<td>4e-19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vbz</td>
<td></td>
<td>2e-9</td>
<td></td>
<td>3e-18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
<td></td>
<td>1e-21</td>
<td>3e-17</td>
<td></td>
</tr>
<tr>
<td>,</td>
<td></td>
<td></td>
<td></td>
<td>4e-20</td>
<td>5e-22</td>
<td></td>
</tr>
</tbody>
</table>
Another Example

<table>
<thead>
<tr>
<th></th>
<th>time</th>
<th>flies</th>
<th>like</th>
<th>an</th>
<th>arrow</th>
<th>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td></td>
<td></td>
<td></td>
<td>10e-15</td>
<td>6e-21</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td></td>
<td></td>
<td>8e-13</td>
<td></td>
<td>1e-19</td>
<td></td>
</tr>
<tr>
<td>jj</td>
<td></td>
<td>6e-14</td>
<td></td>
<td></td>
<td>2e-16</td>
<td></td>
</tr>
<tr>
<td>nn</td>
<td>2e-4</td>
<td></td>
<td></td>
<td>3e-16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>nnp</td>
<td></td>
<td></td>
<td></td>
<td>1e-16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vb</td>
<td>2e-7</td>
<td>1e-14</td>
<td></td>
<td>1e-19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vbp</td>
<td></td>
<td>8e-16</td>
<td></td>
<td>4e-19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vbz</td>
<td></td>
<td>2e-9</td>
<td></td>
<td>3e-18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
<td></td>
<td>1e-21</td>
<td>3e-17</td>
<td></td>
</tr>
<tr>
<td>,</td>
<td></td>
<td></td>
<td></td>
<td>4e-20</td>
<td>5e-22</td>
<td></td>
</tr>
</tbody>
</table>
Formalisms vs. Tasks

• HMMs are a formal concept that apply to a wide range of NLP problems.

• This happens a lot: one tool has many uses.
  – And of course, every problem has a wide range of solutions.

• Three modes in NLP research:
  – Advancing formal models and formalizing interesting ideas. (Risk: may not be relevant to real problems.)
  – Identifying problems with existing approaches and crafting solutions. (Risk: solutions may not generalize.)
  – Identifying new NLP problems to work on and ways to evaluate success.