Information Extraction: Sequence Models, Information Extraction Tasks and Information Integration

Bill MacCartney
CS224N — February 2011
based on slides by Chris Manning
Feature-Based Classifiers

• Exponential (log-linear, maxent, logistic, Gibbs) models:
  – Have features $f_i: C \times D \rightarrow R$, with weights $\lambda_i$, often indicator functions of a condition and class $f_i(c, d) \equiv [\Phi_k(d) \wedge c = c_j]$
  – Use the linear combination $\sum \lambda_i f_i(c, d)$ to produce a probabilistic model:
    \[
    P(c \mid d, \lambda) = \frac{\exp \sum \lambda_i f_i(c, d)}{\sum_{c'} \exp \sum \lambda_i f_i(c', d)}
    \]
    Makes votes positive.
    Normalizes votes.
  – The weights are the parameters of the probability model, combined via a “soft max” function
  – We choose parameters $\{\lambda_i\}$ that maximize the conditional likelihood of the data according to this model.
• Maxent models handle overlapping features well.
• Unlike a NB model, there is no double counting!
Example: NER Overlap

Grace is correlated with PERSON, but does not add much evidence on top of already knowing prefix features.

Local Context

<table>
<thead>
<tr>
<th>Prev</th>
<th>Cur</th>
<th>Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>Other</td>
<td>???</td>
</tr>
<tr>
<td>Word</td>
<td>at</td>
<td>Grace</td>
</tr>
<tr>
<td>Tag</td>
<td>IN</td>
<td>NNP</td>
</tr>
<tr>
<td>Sig</td>
<td>x</td>
<td>Xx</td>
</tr>
</tbody>
</table>

Feature Weights

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>Feature</th>
<th>PERS</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous word</td>
<td>at</td>
<td>-0.73</td>
<td>0.94</td>
</tr>
<tr>
<td>Current word</td>
<td>Grace</td>
<td>0.03</td>
<td>0.00</td>
</tr>
<tr>
<td>Beginning bigram</td>
<td>&lt;G</td>
<td>0.45</td>
<td>-0.04</td>
</tr>
<tr>
<td>Current POS tag</td>
<td>NNP</td>
<td>0.47</td>
<td>0.45</td>
</tr>
<tr>
<td>Prev and cur tags</td>
<td>IN NNP</td>
<td>-0.10</td>
<td>0.14</td>
</tr>
<tr>
<td>Previous state</td>
<td>Other</td>
<td>-0.70</td>
<td>-0.92</td>
</tr>
<tr>
<td>Current signature</td>
<td>Xx</td>
<td>0.80</td>
<td>0.46</td>
</tr>
<tr>
<td>Prev state, cur sig</td>
<td>O-Xx</td>
<td>0.68</td>
<td>0.37</td>
</tr>
<tr>
<td>Prev-cur-next sig</td>
<td>x-Xx-Xx</td>
<td>-0.69</td>
<td>0.37</td>
</tr>
<tr>
<td>P. state - p-cur sig</td>
<td>O-x-Xx</td>
<td>-0.20</td>
<td>0.82</td>
</tr>
<tr>
<td>Total:</td>
<td></td>
<td>-0.58</td>
<td>2.68</td>
</tr>
</tbody>
</table>

Grace is correlated with PERSON, but does not add much evidence on top of already knowing prefix features.
Maxent models handle overlapping features well, but do not automatically model feature interactions.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>All = 1</th>
<th>A = 2/3</th>
<th>B = 2/3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>a</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>B</td>
<td>1/4 1/4</td>
<td>1/3 1/6</td>
<td>4/9 2/9</td>
</tr>
<tr>
<td>b</td>
<td>1/4 1/4</td>
<td>1/3 1/6</td>
<td>2/9 1/9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>b</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Feature Interaction

- If you want interaction terms, you have to add them:

  **Empirical**

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

  **A = 2/3**

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>1/3</td>
<td>1/6</td>
</tr>
<tr>
<td>b</td>
<td>1/3</td>
<td>1/6</td>
</tr>
</tbody>
</table>

  **B = 2/3**

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>4/9</td>
<td>2/9</td>
</tr>
<tr>
<td>b</td>
<td>2/9</td>
<td>1/9</td>
</tr>
</tbody>
</table>

  **AB = 1/3**

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>1/3</td>
<td>1/3</td>
</tr>
<tr>
<td>b</td>
<td>1/3</td>
<td>0</td>
</tr>
</tbody>
</table>

- A disjunctive feature would also have done it (alone):

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>1/3</td>
<td>1/3</td>
</tr>
<tr>
<td>b</td>
<td>1/3</td>
<td>0</td>
</tr>
</tbody>
</table>
Feature Interaction

• For loglinear/logistic regression models in statistics, it is standard to do a greedy stepwise search over the space of all possible interaction terms.
• This combinatorial space is exponential in size, but that’s okay as most statistics models only have 4–8 features.
• In NLP, our models commonly use hundreds of thousands of features, so that’s not okay.
• Commonly, interaction terms are added by hand based on linguistic intuitions.
Example: NER Interaction

Previous-state and current-signature have interactions, e.g. P=PERS-C=Xx indicates C=PERS much more strongly than C=Xx and P=PERS independently.

This feature type allows the model to capture this interaction.

Local Context

<table>
<thead>
<tr>
<th>Prev</th>
<th>Cur</th>
<th>Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>Other</td>
<td>???</td>
</tr>
<tr>
<td>Word</td>
<td>at</td>
<td>Grace</td>
</tr>
<tr>
<td>Tag</td>
<td>IN</td>
<td>NNP</td>
</tr>
<tr>
<td>Sig</td>
<td>x</td>
<td>Xx</td>
</tr>
</tbody>
</table>

Feature Weights

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>Feature</th>
<th>PERS</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous word</td>
<td>at</td>
<td>-0.73</td>
<td>0.94</td>
</tr>
<tr>
<td>Current word</td>
<td>Grace</td>
<td>0.03</td>
<td>0.00</td>
</tr>
<tr>
<td>Beginning bigram</td>
<td>&lt;G</td>
<td>0.45</td>
<td>-0.04</td>
</tr>
<tr>
<td>Current POS tag</td>
<td>NNP</td>
<td>0.47</td>
<td>0.45</td>
</tr>
<tr>
<td>Prev and cur tags</td>
<td>IN NNP</td>
<td>-0.10</td>
<td>0.14</td>
</tr>
<tr>
<td>Previous state</td>
<td>Other</td>
<td>-0.70</td>
<td>-0.92</td>
</tr>
<tr>
<td>Current signature</td>
<td>Xx</td>
<td>0.80</td>
<td>0.46</td>
</tr>
<tr>
<td>Prev state, cur sig</td>
<td>O-Xx</td>
<td>0.68</td>
<td>0.37</td>
</tr>
<tr>
<td>Prev-cur-next sig</td>
<td>x-Xx-Xx</td>
<td>-0.69</td>
<td>0.37</td>
</tr>
<tr>
<td>P. state - p-cur sig</td>
<td>O-x-Xx</td>
<td>-0.20</td>
<td>0.82</td>
</tr>
</tbody>
</table>
... Total: | | -0.58 | 2.68 |
Classification

• What do these joint models of $P(X)$ have to do with conditional models $P(C|D)$?

• Think of the space $C \times D$ as a complex $X$.
  – $C$ is generally small (e.g., 2-100 topic classes)
  – $D$ is generally huge (e.g., space of documents)

• We can, in principle, build models over $P(C,D)$.
• This will involve calculating expectations of features (over $C \times D$):

$$E(f_i) = \sum_{(c,d) \in (C,D)} P(c,d) f_i(c,d)$$

• Generally impractical: can’t enumerate $d$ efficiently.
Classification II

- $D$ may be huge or infinite, but only a few $d$ occur in our data.
- What if we add one feature for each $d$ and constrain its expectation to match our empirical data?

$$\forall (d) \in D \quad P(d) = \hat{P}(d)$$

- Now, most entries of $P(c,d)$ will be zero.
- We can therefore use the much easier sum:

$$E(f_i) = \sum_{(c,d) \in (C,D)} P(c,d) f_i(c,d)$$

$$= \sum_{(c,d) \in (C,D) \land \hat{P}(d) > 0} P(c,d) f_i(c,d)$$
Classification III

• But if we’ve constrained the $D$ marginals

\[
\forall (d) \in D \quad P(d) = \hat{P}(d)
\]

then the only thing that can vary is the conditional distributions:

\[
P(c, d) = P(c \mid d)P(d) = P(c \mid d)\hat{P}(d)
\]

• This is the connection between joint and conditional maxent / exponential models:
  – Conditional models can be thought of as joint models with marginal constraints.

• Maximizing joint likelihood and conditional likelihood of the data in this model are equivalent!
Suppose we train a 1-feature MaxEnt model using the observed data and feature representation shown below.

What is the constructed model’s probability distribution over the four possible outcomes?

A: (i)=0.25 (ii)=0.25 (iii)=0.25 (iv)=0.25
B: (i)=0.33 (ii)=0.16 (iii)=0.33 (iv)=0.16
C: (i)=0.16 (ii)=0.33 (iii)=0.16 (iv)=0.33
D: (i)=0.00 (ii)=0.00 (iii)=0.50 (iv)=0.50
E: None of the above
Issues of Scale

• Lots of features:
  – NLP maxent models can have millions of features.
  – Even storing a single array of parameter values can have a substantial memory cost.

• Lots of sparsity:
  – Overfitting very easy – need smoothing!
  – Many features seen in training will never occur again at test time.

• Optimization problems:
  – Feature weights can be infinite, and iterative solvers can take a long time to get to those infinities.
Smoothing: Issues

- Assume the following empirical distribution:

<table>
<thead>
<tr>
<th></th>
<th>Heads</th>
<th>Tails</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$h$</td>
<td>$t$</td>
</tr>
</tbody>
</table>

- Features: \{Heads\}, \{Tails\}

- We’ll have the following model distribution:

$$ p_{\text{HEADS}} = \frac{e^{\lambda_H}}{e^{\lambda_H} + e^{\lambda_T}} \quad p_{\text{TAILS}} = \frac{e^{\lambda_T}}{e^{\lambda_H} + e^{\lambda_T}} $$

- Really, only one degree of freedom ($\lambda = \lambda_H - \lambda_T$)

$$ p_{\text{HEADS}} = \frac{e^{\lambda_H} e^{-\lambda_T}}{e^{\lambda_H} e^{-\lambda_T} + e^{\lambda_T} e^{-\lambda_T}} = \frac{e^{\lambda}}{e^{\lambda} + e^0} \quad p_{\text{TAILS}} = \frac{e^0}{e^{\lambda} + e^0} $$
**Smoothing: Issues**

- The data likelihood in this model is:

\[
\log P(h, t \mid \lambda) = h \log p_{\text{HEADS}} + t \log p_{\text{TAILS}}
\]
\[
\log P(h, t \mid \lambda) = h\lambda - (t + h) \log (1 + e^{\lambda})
\]
Smoothing: Early Stopping

• In the 4/0 case, there were two problems:
  – The optimal value of $\lambda$ was $\infty$, which is a long trip for an optimization procedure.
  – The learned distribution is just as spiked as the empirical one – no smoothing.

• One way to solve both issues is to just stop the optimization early, after a few iterations.
  – The value of $\lambda$ will be finite (but presumably big).
  – The optimization won’t take forever (clearly).
  – Commonly used in early maxent work.
Smoothing: Priors (MAP)

- What if we had a prior expectation that parameter values wouldn’t be very large?
- We could then balance evidence suggesting large parameters (or infinite) against our prior.
- The evidence would never totally defeat the prior, and parameters would be smoothed (and kept finite!).
- We can do this explicitly by changing the optimization objective to maximum posterior likelihood:

\[
\log P(C, \lambda \mid D) = \log P(\lambda) + \log P(C \mid D, \lambda)
\]

| Posterior | Prior | Evidence |
Smoothing: Priors

- Gaussian, or quadratic, or L2 priors:
  - Intuition: parameters shouldn’t be large.
  - Formalization: prior expectation that each parameter will be distributed according to a gaussian with mean $\mu$ and variance $\sigma^2$.

$$P(\lambda_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left(-\frac{(\lambda_i - \mu_i)^2}{2\sigma_i^2}\right)$$

- Penalizes parameters for drifting to far from their mean prior value (usually $\mu=0$).
- $2\sigma^2 = 1$ works okay.

They don’t even capitalize my name anymore!
Because of smoothing, the more common prefix and single-tag features have larger weights even though entire-word and tag-pair features are more specific.

Local Context

<table>
<thead>
<tr>
<th>State</th>
<th>Prev</th>
<th>Cur</th>
<th>Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word</td>
<td>at</td>
<td>Grace</td>
<td>Road</td>
</tr>
<tr>
<td>Tag</td>
<td>IN</td>
<td>NNP</td>
<td>NNP</td>
</tr>
<tr>
<td>Sig</td>
<td>x</td>
<td>Xx</td>
<td>Xx</td>
</tr>
</tbody>
</table>

Feature Weights

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>Feature</th>
<th>PERS</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous word</td>
<td>at</td>
<td>-0.73</td>
<td>0.94</td>
</tr>
<tr>
<td>Current word</td>
<td>Grace</td>
<td>0.03</td>
<td>0.00</td>
</tr>
<tr>
<td>Beginning bigram</td>
<td>&lt;G</td>
<td>0.45</td>
<td>-0.04</td>
</tr>
<tr>
<td>Current POS tag</td>
<td>NNP</td>
<td>0.47</td>
<td>0.45</td>
</tr>
<tr>
<td>Prev and cur tags</td>
<td>IN NNP</td>
<td>-0.10</td>
<td>0.14</td>
</tr>
<tr>
<td>Previous state</td>
<td>Other</td>
<td>-0.70</td>
<td>-0.92</td>
</tr>
<tr>
<td>Current signature</td>
<td>Xx</td>
<td>0.80</td>
<td>0.46</td>
</tr>
<tr>
<td>Prev state, cur sig</td>
<td>O-Xx</td>
<td>0.68</td>
<td>0.37</td>
</tr>
<tr>
<td>Prev-cur-next sig</td>
<td>x-Xx-Xx</td>
<td>-0.69</td>
<td>0.37</td>
</tr>
<tr>
<td>P. state - p-cur sig</td>
<td>O-x-Xx</td>
<td>-0.20</td>
<td>0.82</td>
</tr>
</tbody>
</table>

Total: -0.58 2.68
Example: POS Tagging

- From (Toutanova et al., 2003):

<table>
<thead>
<tr>
<th></th>
<th>Overall Accuracy</th>
<th>Unknown Word Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without Smoothing</td>
<td>96.54</td>
<td>85.20</td>
</tr>
<tr>
<td>With Smoothing</td>
<td>97.10</td>
<td>88.20</td>
</tr>
</tbody>
</table>

- Smoothing helps:
  - Softens distributions.
  - Pushes weight onto more explanatory features.
  - Allows many features to be dumped safely into the mix.
  - Speeds up convergence (if both are allowed to converge)!
Smoothing: Priors

• If we use gaussian priors:
  – Trade off some expectation-matching for smaller parameters.
  – When multiple features can be recruited to explain a data point, the more common ones generally receive more weight.
  – Accuracy generally goes up!

• Change the objective (assume $\mu_i=0$):

\[
\log P(C, \lambda \mid D) = \log P(C \mid D, \lambda) - \log P(\lambda)
\]

\[
\log P(C, \lambda \mid D) = \sum_{(c,d) \in (C,D)} P(c \mid d, \lambda) - \sum_i \frac{\lambda_i^2}{2\sigma_i^2} + k
\]

• Change the derivative:

\[
\frac{\partial \log P(C, \lambda \mid D)}{\partial \lambda_i} = \text{actual}(f_i, C) - \text{predicted}(f_i, \lambda) - \frac{\lambda_i}{\sigma^2}
\]
Smoothing: Virtual Data

- Another option: smooth the data, not the parameters.
- Example:

  ![Graphs showing smoothing effect](image)

  - Equivalent to adding two extra data points.
  - Similar to add-one smoothing for generative models.
- Hard to know what artificial data to create!
Smoothing: Count Cutoffs

• In NLP, features with low empirical counts were usually dropped.
  – Very weak and indirect smoothing method.
  – Equivalent to locking their weight to be zero.
  – Equivalent to assigning them gaussian priors with mean zero and variance zero.
  – Dropping low counts does remove the features which were most in need of smoothing…
  – … and speeds up the estimation by reducing model size …
  – … but count cutoffs generally hurt accuracy in the presence of proper smoothing.

• We recommend: don’t use count cutoffs unless absolutely necessary.
Sequence Inference

Sequence Level

Sequence Data

Local Level

Local Data

Feature Extraction

Label

Features

Classifier Type

Optimization

Smoothing

Maximum Entropy Models

CG / L-BFGS

Quadratic Penalties

NLP Issues

Inference
MEMM inference in systems

• For a Conditional Markov Model (CMM) a.k.a. a Maximum Entropy Markov Model (MEMM), the classifier makes a single decision at a time, conditioned on evidence from observations and previous decisions.

• A larger space of sequences is explored via search

<table>
<thead>
<tr>
<th>Local Context</th>
<th>Decision Point</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3</td>
<td>-2</td>
<td>-1</td>
</tr>
<tr>
<td>DT</td>
<td>NNP</td>
<td>VBD</td>
</tr>
<tr>
<td>The</td>
<td>Dow</td>
<td>fell</td>
</tr>
<tr>
<td>W₁</td>
<td>%</td>
<td></td>
</tr>
<tr>
<td>W₋₁</td>
<td>fell</td>
<td></td>
</tr>
<tr>
<td>T₋₁</td>
<td>VBD</td>
<td></td>
</tr>
<tr>
<td>T₋₁₋T₋₂</td>
<td>NNP-VBD</td>
<td></td>
</tr>
<tr>
<td>hasDigit?</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td></td>
</tr>
</tbody>
</table>

(Ratnaparkhi 1996; Toutanova et al. 2003, etc.)
Two ways to Search: Beam Inference

- **Beam inference:**
  - At each position keep the top $k$ complete sequences.
  - Extend each sequence in each local way.
  - The extensions compete for the $k$ slots at the next position.

- **Advantages:**
  - Fast; and beam sizes of 3–5 are as good or almost as good as exact inference in many cases.
  - Easy to implement (no dynamic programming required).

- **Disadvantage:**
  - Inexact: the globally best sequence can fall off the beam.
Two ways to Search: Viterbi Inference

• Viterbi inference:
  – Dynamic programming or memoization.
  – Requires small window of state influence (e.g., past two states are relevant).

• Advantage:
  – Exact: the global best sequence is returned.

• Disadvantage:
  – Harder to implement long-distance state-state interactions (but beam inference tends not to successfully capture long-distance resurrection of sequences anyway).
Viterbi Inference: J&M Ch. 6

- I’m basically punting on this … read Ch. 6.
  - I’ll do dynamic programming for parsing
- It’s a small change from HMM Viterbi
  - From:

\[ v_t(j) = \max_{i=1}^{N} v_{t-1}(i) P(s_j|s_i) P(o_t|s_j) \quad 1 \leq j \leq N, 1 < t \leq T \]

  - To:

\[ v_t(j) = \max_{i=1}^{N} v_{t-1}(i) P(s_j|s_i, o_t) \quad 1 \leq j \leq N, 1 < t \leq T \]
Part-of-speech tagging: Generative HMM Tagging Models of Brants 2000

• Highly competitive with other state-of-the art models
• Trigram HMM with smoothed transition probabilities
• Capitalization feature becomes part of the state – each tag state is split into two e.g.
  NN → <NN,cap>,<NN,not cap>
• Suffix features for unknown words

\[
P(w \mid \text{tag}) = P(\text{suffix} \mid \text{tag})(w \mid \text{suffix}) \\
\approx \hat{P}(\text{suffix})\hat{P}(\text{tag} \mid \text{suffix}) / \hat{P}(\text{tag})
\]

\[
\hat{P}(\text{tag} \mid \text{suffix}_n) = \lambda_1 \hat{P}(\text{tag} \mid \text{suffix}_n) + \lambda_2 \hat{P}(\text{tag} \mid \text{suffix}_{n-1}) + ... + \lambda_n \hat{P}(\text{tag})
\]
MEMM Tagging Models -II

- Ratnaparkhi (1996): local distributions are estimated using maximum entropy models
  - Previous two tags, current word, previous two words, next two words, suffix, prefix, hyphenation, and capitalization features for unknown words
- Toutanova et al. (2003)
  - Richer features, bidirectional inference, better smoothing, better unknown word handling

<table>
<thead>
<tr>
<th>Model</th>
<th>Overall Accuracy</th>
<th>Unknown Words</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEMM (Ratn. 1996)</td>
<td>96.63</td>
<td>85.56</td>
</tr>
<tr>
<td>HMM (Brants 2000)</td>
<td>96.7</td>
<td>85.5</td>
</tr>
<tr>
<td>MEMM (T. et al 2003)</td>
<td>97.24</td>
<td>89.04</td>
</tr>
</tbody>
</table>
CRFs [Lafferty, Pereira, and McCallum 2001]

- Another sequence model: Conditional Random Fields (CRFs)
- A whole-sequence conditional model rather than a chaining of local models.

\[ P(c \mid d, \lambda) = \frac{\exp \sum \lambda_i f_i (c, d)}{\sum \exp \sum \lambda_i f_i (c', d)} \]

- The space of \( c \)'s is now the space of sequences
  - But if the features \( f_i \) remain local, the conditional sequence likelihood can still be calculated exactly using dynamic programming
- Training is slower, but CRFs avoid causal-competition biases
- These (or a variant using a max margin criterion) are seen as the state-of-the-art these days, and fairly standardly used
NER Results: CoNLL (2003) NER task

Task: Predict semantic label of each word in text

<table>
<thead>
<tr>
<th></th>
<th>NNP</th>
<th>I-NP</th>
<th>ORG</th>
<th></th>
<th>NNP</th>
<th>I-NP</th>
<th>ORG</th>
<th></th>
<th>NNP</th>
<th>I-NP</th>
<th>ORG</th>
<th></th>
<th>NNP</th>
<th>I-NP</th>
<th>O</th>
<th></th>
<th>NNP</th>
<th>I-NP</th>
<th>PER</th>
<th></th>
<th>NNP</th>
<th>I-NP</th>
<th>PER</th>
<th></th>
<th>NNP</th>
<th>I-NP</th>
<th>ORG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreign</td>
<td></td>
<td></td>
<td></td>
<td>Ministry</td>
<td></td>
<td></td>
<td></td>
<td>spokesman</td>
<td></td>
<td></td>
<td></td>
<td>Shen</td>
<td></td>
<td></td>
<td></td>
<td>Guofang</td>
<td></td>
<td></td>
<td></td>
<td>told</td>
<td></td>
<td></td>
<td></td>
<td>Reuters</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Standard evaluation is per entity, not per token.
NER Results: Discriminative Model

- Increases from better features, a better classification model.

CoNLL 2003 Shared Task: English
NER; entity precision/recall F1
Sequence models? CoNLL 2003 NER shared task

Results on English Devset

![Bar chart showing results for different models on the CoNLL 2003 NER shared task. The x-axis represents different models (MEMM 1st, CRF, MMMN), and the y-axis represents the F1 score.]
CoNLL NER Results: CMM Order

![Graph showing CoNLL NER results for different orders (0th, 1st, 3rd) and categories (Overall, Loc, Misc, Org, Person).]
Using 3 words works significantly better than using only the current word and the previous two or three tags instead! (Toutanova et al. 2003)
Biomedical NER Motivation

• The biomedical world has a huge body of information, which is growing rapidly.

  – MEDLINE, the primary research database serving the biomedical community, currently contains over 12 million abstracts, with 60,000 new abstracts appearing each month.

  – There is also an impressive number of biological databases containing information on genes, proteins, nucleotide and amino acid sequences, including GenBank, Swiss-Prot, and Fly-Base; each contains entries numbering from the thousands to the millions and are multiplying rapidly.
    • Currently, these resources are curated by hand by expert annotators at enormous expense.
Named Entity Recognition

• General NER vs. Biomedical NER

<PER> Christopher Manning </PER> is a professor at <ORG> Stanford University </ORG>, in <LOC> Palo Alto </LOC>.

<RNA> TAR </RNA> independent transactivation by <PROTEIN> Tat </PROTEIN> in cells derived from the <CELL> CNS </CELL> - a novel mechanism of <DNA> HIV-1 gene </DNA> regulation.
Finkel et al. (2004) Results

- BioNLP task – Identify genes, proteins, DNA, RNA, and cell types

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>68.6%</td>
<td>71.6%</td>
<td>70.1%</td>
</tr>
</tbody>
</table>

Precision = \( \frac{tp}{tp + fp} \)

Recall = \( \frac{tp}{tp + fn} \)

\( F1 = \frac{2(\text{precision})(\text{recall})}{(\text{precision} + \text{recall})} \)
Why is this difficult?

• The list of biomedical entities is growing.
  – New genes and proteins are constantly being discovered, so explicitly enumerating and searching against a list of known entities is not scalable.
  – Part of the difficulty lies in identifying previously unseen entities based on contextual, orthographic, and other clues.

• Biomedical entities don’t have strict naming conventions.
  – Common English words such as period, curved, and for are used for gene names.
  – Entity names can be ambiguous. For example, in FlyBase, “clk” is the gene symbol for the “Clock” gene but it also is used as a synonym of the “period” gene.

• Biomedical entity names are ambiguous
  – Experts only agree on whether a word is even a gene or protein 69% of the time! (Krauthammer et al., 2000)
  – Often systematic polysemies between gene, RNA, DNA, etc.
Features: What’s in a Name?

- oxa
- : 708
- field

- drug
- company
- movie
- place
- person

Cotrimoxazole

Alien Fury: Countdown to Invasion
Interesting Features

- Word, and surrounding context
- Word Shapes
  - Map words to simplified representation that encodes attributes such as length, capitalization, numerals, Greek letters, internal punctuation, etc.

<table>
<thead>
<tr>
<th>Term</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Varicella-zoster</td>
<td>Xx-xxx</td>
</tr>
<tr>
<td>mRNA</td>
<td>xXXX</td>
</tr>
<tr>
<td>CPA1</td>
<td>XXXd</td>
</tr>
</tbody>
</table>

- Character substrings
The full task of “Information Extraction”

As a family of techniques:

Information Extraction = segmentation + classification + association + clustering

October 14, 2002, 4:00 a.m. PT

For years, Microsoft Corporation CEO Bill Gates railed against the economic philosophy of open-source software with Orwellian fervor, denouncing its communal licensing as a "cancer" that stifled technological innovation.

Today, Microsoft claims to "love" the open-source concept, by which software code is made public to encourage improvement and development by outside programmers. Gates himself says Microsoft will gladly disclose its crown jewels—the coveted code behind the Windows operating system—to select customers.

"We can be open source. We love the concept of shared source," said Bill Veghte, a Microsoft VP. "That's a super-important shift for us in terms of code access."

Richard Stallman, founder of the Free Software Foundation, countered saying...
May 19 1995 Atlanta -- The Centers for Disease Control and Prevention, which is in the front line of the world's response to the deadly Ebola epidemic in Zaire, is finding itself hard pressed to cope with the crisis...

Disease Outbreaks in *The New York Times*

<table>
<thead>
<tr>
<th>Date</th>
<th>Disease Name</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan. 1995</td>
<td>Malaria</td>
<td>Ethiopia</td>
</tr>
<tr>
<td>July 1995</td>
<td>Mad Cow Disease</td>
<td>U.K.</td>
</tr>
<tr>
<td>Feb. 1995</td>
<td>Pneumonia</td>
<td>U.S.</td>
</tr>
</tbody>
</table>

- Can be done by hand-written rules over text, perhaps annotated for NER, POS.
- Commonly done as classification decision based on identified entities and words/parse tree patterns between them.
- Considerable work on bootstrapping learning from seed examples.
Landscape of IE Tasks (1/4): Degree of Formatting

Text paragraphs without formatting

Astro Teller is the CEO and co-founder of BodyMedia. Astro holds a Ph.D. in Artificial Intelligence from Carnegie Mellon University, where he was inducted as a national Hertz fellow. His M.S. in symbolic and heuristic computation and B.S. in computer science are from Stanford University. His work in science, literature and business has appeared in international media from the New York Times to CNN to NPR.

Grammatical sentences and some formatting & links

Dr. Steven Minton - Founder/CTO
Dr. Minton is a fellow of the American Association of Artificial Intelligence and was the founder of the Journal of Artificial Intelligence Research. Prior to founding Fetch, Minton was a faculty member at USC and a project leader at USC's Information Sciences Institute. A graduate of Yale University and Carnegie Mellon University, Minton has been a Principal Investigator at NASA Ames and taught at Stanford, UC Berkeley and USC.

Frank Huybrechts - COO
Mr. Huybrechts has over 20 years of

Non-grammatical snippets, rich formatting & links

Barto, Andrew G.  (413) 545-2109  barto@cs.umass.edu  CS276
Professor.
Computational neuroscience, reinforcement learning, adaptive motor control, artificial neural networks, adaptive and learning control, motor development.

Berger, Emery D.  (413) 577-4211  emery@cs.umass.edu  CS344
Assistant Professor.

Brock, Oliver  (413) 577-0334  oli@cs.umass.edu  CS246
Assistant Professor.

Clarke, Lori A.  (413) 545-1328  clarke@cs.umass.edu  CS304
Professor.
Software verification, testing, and analysis; software architecture and design.

Cohen, Paul R.  (413) 545-3638  cohen@cs.umass.edu  CS278
Professor.
Planning, simulation, natural language, agent-based systems, intelligent data analysis, intelligent user interfaces.

Tables

8:30 - 9:30 AM  Invited Talk: Plausibility Measures: A General Approach for Representing Uncertainty
Joseph Y. Halpern, Cornell University

9:30 - 10:00 AM  Coffee Break

10:00 - 11:30 AM  Technical Paper Sessions:

Cognitive Robotics  Logic Programming  Natural Language Generation  Complexity Analysis  Neural Networks  Games

739: A Logical Account of Causal and Topological Maps
Emilio Remolina and Benjamin Kupsers

116: A System: Problem Solving through Abduction
Marc Denecker, Antonis Kakas, and Bert Van Nuffelen

788: Title Generation for Machine-Translated Documents
Rong Jin and Alexander G. Hauptmann

417: Let's go Nats: Complexity of Nested Circumscription and Abnormality Theories
Marco Cadoli, Thomas Eiter, and Georg Gottlob

179: Knowledge Extraction and Comparison from Local Function Networks
Kenneth McGregor, Stefan Weimer, and John MacIntyre

71: Iterative Widening
Tristan Carew

549: Online-Execution of eGeolg Plans
Henrik Grosskreutz and Gerhard Lakemeyer

131: A Comparative Study of Logic Programs with Preference
Torsten Schaub and Reven

246: Dealing with Dependencies between Content Planning and Surface Realization in a Pipeline Generation

470: A Perspective on Knowledge Compilation
Adton Darwiche and Pierre Marquis

258: Violation-Guided Learning of Formulations in Neural-Network Time-Series

353: Temporal Difference Learning
Applied to a High Performance Game-Playing
Landscape of IE Tasks (2/4): Intended Breadth of Coverage

Web site specific
- Formatting
  - Amazon.com Book Pages

Genre specific
- Layout
  - Resumes

Wide, non-specific
- Language
  - University Names
Landscape of IE Tasks (3/4): Complexity

E.g. word patterns:

**Closed set**
- U.S. states
- He was born in Alabama…
- The big Wyoming sky…

**Regular set**
- U.S. phone numbers
- Phone: (413) 545-1323
- The CALD main office can be reached at 412-268-1299

**Complex pattern**
- U.S. postal addresses
- University of Arkansas
  P.O. Box 140
  Hope, AR 71802
- Headquarters:
  1128 Main Street, 4th Floor
  Cincinnati, Ohio 45210

**Ambiguous patterns, needing context and many sources of evidence**
- Person names
  - …was among the six houses sold by Hope Feldman that year.
  - Pawel Opalinski, Software Engineer at WhizBang Labs.
Jack Welch will retire as CEO of General Electric tomorrow. The top role at the Connecticut company will be filled by Jeffrey Immelt.

**Single entity**
- **Person**: Jack Welch
- **Location**: Connecticut

**Binary relationship**
- **Relation**: Person-Title
  - **Person**: Jack Welch
  - **Title**: CEO

**N-ary record**
- **Relation**: Succession
  - **Company**: General Electric
  - **Title**: CEO
  - **Out**: Jack Welch
  - **In**: Jeffrey Immelt

“Named entity” extraction