MACHINE LEARNING FOR NATURAL LANGUAGE PROCESSING
Outline

• Some Sample NLP Task [Noah Smith]
• Structured Prediction For NLP
• Structured Prediction Methods
  – Conditional Random Fields
  – Structured Perceptron
• Discussion
MOTIVATING STRUCTURED-OUTPUT PREDICTION FOR NLP
Types of machine learning

Classifier
Prediction of categorical output or class
One of a few discrete values

Density Estimator
Probability

Regressor
Prediction of real-valued output
E.g., (Part of Speech) Tagging

Bill directed plays about English kings
E.g., Segmentation into Words

第二阶段的奥运会体育比赛门票与残奥会开
闭幕式门票的预订工作已经结束, 现在进入
门票分配阶段。在此期间, 我们不再接受新的
门票预订申请。
E.g., Segmentation within Words

uygarlaştıramadıklarımızdanmışınızıcına

“(behaving) as if you are among those whom we could not civilize”
Britain sent warships across the English Channel on Monday to rescue Britons stranded by Eyjafjallajökull's volcanic ash cloud.
E.g., Trees

Britain sent warships across the English Channel on Monday to rescue Britons stranded by Eyjafjallajökull's volcanic ash cloud.
E.g., Predicate-Argument Structures

Britain sent warships across the English Channel

Monday to rescue Britons stranded by Eyjafjallajökull 's volcanic ash cloud
Mr President, Noah's ark was filled not with
Noahs Arche war nicht voller

production factors, but with living creatures.

Produktionsfaktoren, sondern Geschöpfe.
Outline

• Some Sample NLP Task  [Noah Smith]
  – What do these have in common?
  – We’re making many related predictions ...

• Structured Prediction For NLP
• Structured Prediction Methods
  – Structured Perceptron
  – Conditional Random Fields
• Deep learning and NLP
Britain sent warships across the English Channel on Monday to rescue Britons stranded by Eyjafjallajökull's volcanic ash cloud.
E.g., Segmentation *and* Tagging

via begin-inside-outside **token tagging**

Britain sent warships across the **English Channel**

**geopolitical entity**

**beginGPE**

**out**

**out**

**out**

**out**

**beginGF**

**inGF**

**geographic feature**
E.g., Segmentation *and* Tagging via begin-inside-outside token tagging

**beginGF**

**inGF**

**via** begin-inside-outside **token tagging**

Britain sent warships across the English Channel

**beginX** inX* delimits entity X

- thisToken = english
- thisTokenShape = Xx+
- prevToken = the
- prevPrevToken = across
- nextToken = channel
- nextNextToken = Monday
- nextTokenShape = Xx+
- prevTokenShape = x+
- ....

- thisToken_english = 1
- thisTokenShape_Xx+ = 1
- prevToken_the = 1
- prevPrevToken_across = 1
- ...

**a classification task!**
E.g., Segmentation and Tagging via begin-inside-outside token tagging

A problem: the instances are not i.i.d, nor are the classes.

Britain sent warships across the English Channel

- thisToken_the = 1
- thisTokenShape_x+ = 1
- prevToken_across = 1
- ...

- thisToken_english = 1
- thisTokenShape_Xx+ = 1
- prevToken_the = 1
- prevPrevToken_across = 1
- ...

- thisToken_channel = 1
- thisTokenShape_Xx+ = 1
- prevToken_english = 1
- prevPrevToken_the = 1
- ...

out

beginGF

inGF
Structured Output Prediction

• Structured Output Prediction: definition
  – classifier where output is structured, and input might be structured
  – example: predict a sequence of begin-in-out tags from a sequence of tokens (represented by a sequence of feature vectors)
NER via begin-inside-outside token tagging

A problem: with \(K\) begin-inside-outside tags and \(N\) words, we have \(N^K\) possible structured labels.

How can we learn to choose among so many possible outputs?

\[ x = \text{\textcolor{red}{Britain} sent \textcolor{green}{warships} across the \textcolor{blue}{English Channel}} \]

\[ y = \begin{smallmatrix}
\text{beginGPE} & \text{out} & \text{out} & \text{out} & \text{out} & \text{beginGF} & \text{inGF}
\end{smallmatrix} \]
HMMS AND NER
e.g., HMMs map a sequence of tokens to a sequence of outputs (the hidden states)

\[ x = \text{[Britain sent warships across the English Channel]} \]

\[ y = \{ \text{beginGPE} \text{ out} \text{ out} \text{ out} \text{ out} \text{ beginGF} \text{ inGF} \} \]
Borkar et al’s: HMMs for segmentation

- Example: Addresses, bib records
- Problem: some DBs may split records up differently (eg no “mail stop” field, combine address and apt #, …) or not at all
- Solution: Learn to segment textual form of records

IE with Hidden Markov Models

- Must learn transition, emission probabilities.
HMMS limitation: not well-suited to modeling a set of features for each token.

Britain sent warships across the English Channel

- thisToken = english
- thisTokenShape = Xx+
- prevToken = the
- prevPrevToken = across
- nextToken = channel
- nextNextToken = Monday
- nextTokenShape = Xx+
- prevTokenShape = x+
- ...

- thisToken_english = 1
- thisTokenShape_Xx+ = 1
- prevToken_the = 1
- prevPrevToken_across = 1
- ...

a classification task!
What is a symbol?

Ideally we would like to use many, arbitrary, overlapping features of words.

identity of word
ends in “-ski”
is capitalized
is part of a noun phrase
is in a list of city names
is under node X in WordNet
is in bold font
is indented
is in hyperlink anchor
...

Lots of learning systems are not confounded by multiple, non-independent features: decision trees, neural nets, SVMs, …
Outline

• Some Sample NLP Task  [Noah Smith]
• Structured Prediction For NLP
• Structured Prediction Methods
  – HMMs
  – Conditional Random Fields
  – Structured Perceptron
• Discussion
CONDITIONAL RANDOM FIELDS
What is a symbol?

Ideally we would like to use many, arbitrary, overlapping features of words.

identity of word
ends in “-ski”
is capitalized
is part of a noun phrase
is in a list of city names
is under node X in WordNet
is in bold font
is indented
is in hyperlink anchor

Lots of learning systems are not confounded by multiple, non-independent features: decision trees, neural nets, SVMs, …
Inference for linear-chain MRFs

When will prof Cohen post the notes ...

Idea 1: features are properties of two adjacent tokens, and the pair of labels assigned to them.

• \((y(i) == \text{B} \text{ or } y(i) == \text{I}) \text{ and } \text{(token}(i)\text{ is capitalized})\)

• \((y(i) == \text{I} \text{ and } y(i-1) == \text{B}) \text{ and } \text{(token}(i)\text{ is hyphenated})\)

• \((y(i) == \text{B} \text{ and } y(i-1) == \text{B})\)

  • eg “tell Ziv William is on the way”

Idea 2: construct a graph where each path is a possible sequence labeling.
Inference for a linear-chain MRF

When will prof Cohen post the notes ...

• Inference: find the highest-weight path
• Learning: optimize the feature weights so that this highest-weight path is correct relative to the data
Feature + weights define edge weights:

- \( \text{weight}(y_i, y_{i+1}) = 2 \times [(y_i = B \text{ or } I) \text{ and } \text{isCap}(x_i)] + 1 \times [(y_i = B \text{ and } \text{isFirstName}(x_i)] - 5 \times [(y_{i+1} \neq B \text{ and } \text{isLower}(x_i) \text{ and } \text{isUpper}(x_{i+1})] \)
The CRF’s global feature vector for input sequence $x$ and label sequence $y$ is given by

$$F(y, x) = \sum_i f(y, x, i)$$

where $i$ ranges over input positions. The conditional probability distribution defined by the CRF is then

$$p_\lambda(Y | X) = \frac{\exp \lambda \cdot F(Y, X)}{Z_\lambda(X)} \quad (1)$$

where

$$Z_\lambda(x) = \sum_y \exp \lambda \cdot F(y, x)$$
CRF learning – from Sha & Pereira

The most probable label sequence for input sequence $x$ is

$$\hat{y} = \arg \max_y p_{\lambda}(y|x) = \arg \max_y \lambda \cdot F(y, x)$$

because $Z_{\lambda}(x)$ does not depend on $y$. $F(y, x)$ decomposes into a sum of terms for consecutive pairs of labels, so the most likely $y$ can be found with the Viterbi algorithm.

We train a CRF by maximizing the log-likelihood of a given training set $T = \{(x_k, y_k)\}_{k=1}^N$, which we assume fixed for the rest of this section:

$$\mathcal{L}_{\lambda} = \sum_k \log p_{\lambda}(y_k|x_k)$$

$$= \sum_k [\lambda \cdot F(y_k, x_k) - \log Z_{\lambda}(x_k)]$$

To perform this optimization, we seek the zero of the gradient

$$\nabla \mathcal{L}_{\lambda} =$$

$$\sum_k \left[ F(y_k, x_k) - E_{p_{\lambda}(Y|x_k)} F(Y, x_k) \right]$$

(2)
CRF learning – from Sha & Pereira

To perform this optimization, we seek the zero of the gradient

$$\nabla \mathcal{L}_\lambda = \sum_k \left[ F(y_k, x_k) - E_{p^\lambda(Y|x_k)} F(Y, x_k) \right]$$  \hspace{1cm} (2)$$

Something like forward-backward

Idea:

• Define matrix of y,y’ “affinities” at stage i

• $M_i[y,y'] = “unnormalized probability”$ of transition from y to y’ at stage i

• $M_i * M_{i+1} = “unnormalized probability”$ of any path through stages i and i+1
Sha & Pereira results

<table>
<thead>
<tr>
<th>$q(y_{i-1}, y_i)$</th>
<th>$p(x, t)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_i = y$</td>
<td>$w_i = w$</td>
</tr>
<tr>
<td>$y_i = y$, $y_{i-1} = y'$</td>
<td>$w_{i-1} = w$</td>
</tr>
<tr>
<td>$c(y_i) = c$</td>
<td>$w_{i+1} = w$</td>
</tr>
<tr>
<td></td>
<td>$w_{i-2} = w$</td>
</tr>
<tr>
<td></td>
<td>$w_{i+2} = w$</td>
</tr>
<tr>
<td></td>
<td>$w_{i-1} = w'$, $w_i = w$</td>
</tr>
<tr>
<td></td>
<td>$w_{i+1} = w'$, $w_i = w$</td>
</tr>
<tr>
<td></td>
<td>$t_i = t$</td>
</tr>
<tr>
<td></td>
<td>$i_{i-1} = t$</td>
</tr>
<tr>
<td></td>
<td>$i_{i+1} = t$</td>
</tr>
<tr>
<td></td>
<td>$i_{i-2} = t$</td>
</tr>
<tr>
<td></td>
<td>$i_{i+2} = t$</td>
</tr>
<tr>
<td></td>
<td>$t_{i-1} = t', t_i = t$</td>
</tr>
<tr>
<td></td>
<td>$t_{i-2} = t'$, $t_{i-1} = t$</td>
</tr>
<tr>
<td></td>
<td>$t_i = t'$, $t_{i+1} = t$</td>
</tr>
<tr>
<td></td>
<td>$t_{i+1} = t'$, $t_{i+2} = t$</td>
</tr>
<tr>
<td></td>
<td>$t_{i-2} = t''$, $t_{i-1} = t'$, $t_i = t$</td>
</tr>
<tr>
<td></td>
<td>$t_{i-1} = t''$, $t_i = t'$, $t_{i+1} = t$</td>
</tr>
<tr>
<td></td>
<td>$t_i = t''$, $t_{i+1} = t'$, $t_{i+2} = t$</td>
</tr>
</tbody>
</table>

Table 1: Shallow parsing features

<table>
<thead>
<tr>
<th>Model</th>
<th>F score</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM combination</td>
<td>94.39%</td>
</tr>
<tr>
<td>(Kudo and Matsumoto, 2001)</td>
<td></td>
</tr>
<tr>
<td>CRF</td>
<td>94.38%</td>
</tr>
<tr>
<td>Generalized winnow</td>
<td>93.89%</td>
</tr>
<tr>
<td>(Zhang et al., 2002)</td>
<td></td>
</tr>
<tr>
<td>Voted perceptron</td>
<td>94.09%</td>
</tr>
<tr>
<td>MEMM</td>
<td>93.70%</td>
</tr>
</tbody>
</table>

Table 2: NP chunking F scores

CRF beats MEMM (McNemar’s test); MEMM probably beats voted perceptron
Sha & Pereira results

<table>
<thead>
<tr>
<th>training method</th>
<th>time</th>
<th>F score</th>
<th>$L'_\lambda$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precond. CG</td>
<td>130</td>
<td>94.19%</td>
<td>-2968</td>
</tr>
<tr>
<td>Mixed CG</td>
<td>540</td>
<td>94.20%</td>
<td>-2990</td>
</tr>
<tr>
<td>Plain CG</td>
<td>648</td>
<td>94.04%</td>
<td>-2967</td>
</tr>
<tr>
<td>L-BFGS</td>
<td>84</td>
<td>94.19%</td>
<td>-2948</td>
</tr>
<tr>
<td>GIS</td>
<td>3700</td>
<td>93.55%</td>
<td>-5668</td>
</tr>
</tbody>
</table>

Table 3: Runtime for various training methods in minutes, 375k examples
THE STRUCTURED PERCEPTRON
Review: The perceptron

**instance** $x_i$

Compute: $\hat{y}_i = \mathbf{v}_k \cdot \mathbf{x}_i$

If mistake: $\mathbf{v}_{k+1} = \mathbf{v}_k + y_i \mathbf{x}_i$

**Margin** $\gamma$. $A$ must provide examples that can be separated with some vector $\mathbf{u}$ with margin $\gamma > 0$, ie

$$\exists \mathbf{u}: \forall (x_i, y_i) \text{ given by } A, (\mathbf{u} \cdot \mathbf{x})y_i > \gamma$$

and furthermore, $\|\mathbf{u}\| = 1$.

**Radius** $R$. $A$ must provide examples “near the origin”, ie

$$\forall x_i \text{ given by } A, \|x\|^2 < R$$
Lemma 1  \( \forall k, \mathbf{v}_k \cdot \mathbf{u} \geq k\gamma \). In other words, the dot product between \( \mathbf{v}_k \) and \( \mathbf{u} \) increases with each mistake, at a rate depending on the margin \( \gamma \).

Proof:

\[
\mathbf{v}_{k+1} \cdot \mathbf{u} = (\mathbf{v}_k + y_i \mathbf{x}_i) \cdot \mathbf{u} \\
\Rightarrow \mathbf{v}_{k+1} \cdot \mathbf{u} = (\mathbf{v}_k \cdot \mathbf{u}) + y_i (\mathbf{x}_i \cdot \mathbf{u}) \\
\Rightarrow \mathbf{v}_{k+1} \cdot \mathbf{u} \geq \mathbf{v}_k \cdot \mathbf{u} + \gamma \\
\Rightarrow \mathbf{v}_k \cdot \mathbf{u} \geq k\gamma
\]
(3a) The guess $v_2$ after the two positive examples: $v_2 = v_1 + x_2$

(3b) The guess $v_2$ after the one positive and one negative example: $v_2 = v_1 - x_2$

Lemma 2 $\forall k, \|v_k\|^2 \leq kR$. In other words, the norm of $v_k$ grows “slowly”, at a rate depending on $R$.

Proof:

$$v_{k+1} \cdot v_{k+1} = (v_k + y_i x_i) \cdot (v_k + y_i x_i)$$

$$\Rightarrow \quad \|v_{k+1}\|^2 = \|v_{k+1}\|^2 + 2y_i x_i \cdot v_k + y_i^2 x \cdot x$$

$$\Rightarrow \quad \|v_{k+1}\|^2 = \|v_{k+1}\|^2 + [\text{something negative}] + 1\|x\|^2$$

$$\Rightarrow \quad \|v_{k+1}\|^2 \leq \|v_{k+1}\|^2 + \|x\|^2$$

$$\Rightarrow \quad \|v_{k+1}\|^2 \leq \|v_{k+1}\|^2 + R$$

$$\Rightarrow \quad \|v_k\|^2 \leq kR$$
Lemma 1 \( \forall k, v_k \cdot u \geq k\gamma \). In other words, the dot product between \( v_k \) and \( u \) increases with each mistake, at a rate depending on the margin \( \gamma \).

Lemma 2 \( \forall k, \| v_k \|^2 \leq kR \). In other words, the norm of \( v_k \) grows “slowly”, at a rate depending on \( R \).

\[
(k\gamma)^2 \leq (v_k \cdot u)^2
\]
\[
\Rightarrow k^2\gamma^2 \leq \| v_k \|^2 \| u \|^2
\]
\[
\Rightarrow k^2\gamma^2 \leq \| v_k \|^2
\]

\[
k^2\gamma^2 \leq \| v_k \|^2 \leq kR^2
\]
\[
\Rightarrow k^2\gamma^2 \leq kR^2
\]
\[
\Rightarrow k\gamma^2 \leq R^2
\]
\[
\Rightarrow k \leq \frac{R^2}{\gamma^2} = \left( \frac{R}{\gamma} \right)^2
\]

Radius \( R \). A must provide examples “near the origin”, ie

\[
\forall x_i \text{ given by } A, \| x \|^2 < R^2
\]
The voted perceptron for ranking

Instances $x_1 x_2 x_3 x_4 \ldots$

Compute: $y_i = \mathbf{v}_k \cdot \mathbf{x}_i$

Return: the index $b^*$ of the "best" $\mathbf{x}_i$

If mistake: $\mathbf{v}_{k+1} = \mathbf{v}_k + \mathbf{x}_b - \mathbf{x}_{b^*}$

Margin $\gamma$. A must provide examples that can be correctly ranked with some vector $\mathbf{u}$ with margin $\gamma > 0$, i.e.

$\exists \mathbf{u} : \forall x_{i,1}, \ldots, x_{i,n_i}, \ell$ given by $A$, $\forall j \neq \ell$, $\mathbf{u} \cdot x_\ell - \mathbf{u} \cdot x_j > \gamma$

and furthermore, $\|\mathbf{u}\|^2 = 1$.

Radius $R$. A must provide examples "near the origin", i.e.

$\forall x_i$ given by $A$, $\|x\|^2 < R$
Ranking some $x$'s with the target vector $u$. 
Ranking some x’s with some guess vector $v$ – part 1
Ranking some $x$’s with some guess vector $v$ – part 2.

The purple-circled $x$ is $x_{b^*}$ - the one the learner has chosen to rank highest. The green circled $x$ is $x_b$, the right answer.
Correcting \( \mathbf{v} \) by adding \( x_b - x_{b^*} \)
Correcting $\mathbf{v}$ by adding $x_b - x_{b^*}$

(part 2)
The guess $v_2$ after the two positive examples: $v_2 = v_1 + x_2$

Lemma 1 \( \forall k, \ v_k \cdot u \geq k\gamma \). In other words, the dot product between $v_k$ and $u$ increases with each mistake, at a rate depending on the margin $\gamma$.

Proof:

\[
\begin{align*}
v_{k+1} \cdot u &= (v_k + y_i x_i) \cdot u \\
\Rightarrow \quad v_{k+1} \cdot u &= (v_k \cdot u) + y_i (x_i \cdot u) \\
\Rightarrow \quad v_{k+1} \cdot u &\geq v_k \cdot u + \gamma \\
\Rightarrow \quad v_k \cdot u &\geq k\gamma
\end{align*}
\]
The guess $v_2$ after the two positive examples: $v_2 = v_1 + x_2$

Lemma 3 \( \forall k, \; v_k \cdot u \geq k\gamma \). In other words, the dot product between $v_k$ and $u$ increases with each mistake, at a rate depending on the margin $\gamma$.

\[
\begin{align*}
v_{k+1} \cdot u &= (v_k + y_i x_i) \cdot u \\
\Rightarrow v_{k+1} \cdot u &= (v_k \cdot u) + y_i (x_i \cdot u) \\
\Rightarrow v_{k+1} \cdot u &\geq v_k \cdot u + \gamma \\
\Rightarrow v_k \cdot u &\geq k\gamma 
\end{align*}
\]
The guess $v_2$ after the two positive examples: $v_2 = v_1 + x_2$

Lemma 3 \( \forall k, v_k \cdot u \geq k\gamma \). In other words, the dot product between $v_k$ and $u$ increases with each mistake, at a rate depending on the margin $\gamma$.

\[
\begin{align*}
    v_{k+1} \cdot u &= (v_k + y_i x_i) \cdot u \\
    \Rightarrow v_{k+1} \cdot u &= (v_k \cdot u) + y_i (x_i \cdot u) \\
    \Rightarrow v_{k+1} \cdot u &\geq v_k \cdot u + \gamma \\
    \Rightarrow v_k \cdot u &\geq k\gamma
\end{align*}
\]

\[
\begin{align*}
    v_{k+1} \cdot u &= (v_k + x_{i,\ell} - x_{i,\ell}) \cdot u \\
    \Rightarrow v_{k+1} \cdot u &= v_k \cdot u + x_{i,\ell} \cdot u - x_{i,\ell} \cdot u \\
    \Rightarrow v_{k+1} \cdot u &\geq v_k \cdot u + \gamma \\
    \Rightarrow v_k \cdot u &\geq k\gamma
\end{align*}
\]
Notice this doesn’t depend at all on the number of x’s being ranked

(3a) The guess $v_2$ after the two positive examples: $v_2 = v_1 + x_2$

Lemma 4 $\forall k, \|v_k\|^2 \leq 2kR$.

Theorem 2 Under the rules of the ranking perceptron game, it is always the case that $k < 2R/\gamma^2$.

Neither proof depends on the dimension of the x’s.
The voted perceptron for ranking

\[ y_i = \mathbf{v}_k \cdot \mathbf{x}_i \]

Return: the index \( b^* \) of the “best” \( \mathbf{x}_i \)

If mistake:
\[ \mathbf{v}_{k+1} = \mathbf{v}_k + \mathbf{x}_b - \mathbf{x}_{b^*} \]

Change number one is notation: replace \( \mathbf{x} \) with \( \mathbf{g} \)
The voted perceptron for NER

Compute: \( y_i = \hat{v}_k \cdot g_i \)

Return: the index \( b_* \) of the “best” \( g_i \)

If mistake: \( v_{k+1} = v_k + g_b - g_{b_*} \)

1. A sends B the Sha & Pereira paper, some feature functions, and instructions for creating the instances \( g \):
   - A sends a word vector \( x_i \). Then B could create the instances \( g_1 = F(x_i, y_1), g_2 = F(x_i, y_2), ... \)
   - but instead B just returns the \( y* \) that gives the best score for the dot product \( v_k \cdot F(x, y*) \) by using Viterbi.

2. A sends B the correct label sequence \( y_i \).

3. On errors, B sets \( v_{k+1} = v_k + g_b - g_{b_*} = v_k + F(x, y) - F(x, y^*) \)
To perform this optimization, we seek the zero of the gradient

\[ \nabla \mathcal{L}_\lambda = \sum_k \left[ F(y_k, x_k) - E_{p_\lambda(Y|x_k)} F(Y, x_k) \right] \]  

(2)

1. A sends a word vector \( x_i \).
2. B just returns the \( y^* \) that gives the best score for \( v_k \cdot F(x_i, y^*) \).
3. A sends B the correct label sequence \( y_i \).
4. On errors, B sets \( v_{k+1} = v_k + z_b - z_{b^*} = v_k + F(x_i, y) - F(x_i, y^*) \).

So, this algorithm can also be viewed as an approximation to the CRF learning algorithm – where we’re using a Viterbi approximation to the expectations, and stochastic gradient descent to optimize the likelihood.
And back to the paper.....


Michael Collins
AT&T Labs-Research, Florham Park, New Jersey.
mcollins@research.att.com

EMNLP 2002, Best paper
Collins’ Experiments

- POS tagging (with MXPOST features)
- NP Chunking (words and POS tags from Brill’s tagger as features) and BIO output tags
- Compared Maxent Tagging/MEMM’s (with iterative scaling) and “Voted Perceptron trained HMM’s”
  - With and w/o averaging
  - With and w/o feature selection (count>5)
Collins’ results

<table>
<thead>
<tr>
<th>Method</th>
<th>F-Measure</th>
<th>Numits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perc, avg, cc=0</td>
<td>93.53</td>
<td>13</td>
</tr>
<tr>
<td>Perc, noavg, cc=0</td>
<td>93.04</td>
<td>35</td>
</tr>
<tr>
<td>Perc, avg, cc=5</td>
<td>93.33</td>
<td>9</td>
</tr>
<tr>
<td>Perc, noavg, cc=5</td>
<td>91.88</td>
<td>39</td>
</tr>
<tr>
<td>ME, cc=0</td>
<td>92.34</td>
<td>900</td>
</tr>
<tr>
<td>ME, cc=5</td>
<td>92.65</td>
<td>200</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Error rate/%</th>
<th>Numits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perc, avg, cc=0</td>
<td>2.93</td>
<td>10</td>
</tr>
<tr>
<td>Perc, noavg, cc=0</td>
<td>3.68</td>
<td>20</td>
</tr>
<tr>
<td>Perc, avg, cc=5</td>
<td>3.03</td>
<td>6</td>
</tr>
<tr>
<td>Perc, noavg, cc=5</td>
<td>4.04</td>
<td>17</td>
</tr>
<tr>
<td>ME, cc=0</td>
<td>3.4</td>
<td>100</td>
</tr>
<tr>
<td>ME, cc=5</td>
<td>3.28</td>
<td>200</td>
</tr>
</tbody>
</table>

Figure 4: Results for various methods on the part-of-speech tagging and chunking tasks on development data. All scores are error percentages. Numits is the number of training iterations at which the best score is achieved. Perc is the perceptron algorithm, ME is the maximum entropy method. Avg/noavg is the perceptron with or without averaged parameter vectors. cc=5 means only features occurring 5 times or more in training are included, cc=0 means all features in training are included.
Outline

- Some Sample NLP Task  [Noah Smith]
- Structured Prediction For NLP
- Structured Prediction Methods
  - Conditional Random Fields
  - Structured Perceptron
- Discussion
Discussion

• Structured prediction is just one part of NLP
  – There is *lots* of work on learning and NLP
  – It would be much easier to summarize non-learning related NLP

• CRFs/HMMs/structured perceptrons are just one part of structured output prediction
  – Chris Dyer course …. 

• Hot topic now: representation learning and deep learning for NLP
  • NAACL 2013: Deep learning for NLP, Manning