LING 581: Advanced Computational Linguistics

Lecture Notes
February 12th
Today's Topics

1. Passives in the WSJ PTB
2. MXPOST
3. BIKEL COLLINS
4. EVALB
How many passives?

• Passives:

```
7739: @S < ~/^NP.*-SBJ-([0-9]+)/#1%i <= ( VP < (VBN  
  .. (@NP < ( -NONE- < /*-([0-9]+)/#1%i)))))
```
How many passives?

• Example: -TTL (title)
How many passives?

• Example:

- Rudolph Agnew, 55 years old and former chairman of Consolidated Gold Fields PLC, was named a nonexecutive director of this British industrial conglomerate.

10946: @S < /^NP.*-SBJ-([0-9]+)/#1%i << ( VP < VBN << (@NP < ( -NONE- < /\*-([0-9]+)/#1%i)) )
How many passives?

• Example (false positive):
  – New England Electric, based in Westborough, Mass., had offered $2 billion to acquire PS of New Hampshire, well below the $2.29 billion value United Illuminating places on its bid and the $2.25 billion Northeast says its bid is worth.
How many passives?

- Need some form of the verb *be*:
  - 8048: @S < /^NP.*-SBJ-([0-9]+)#1%i << ( VP < (/^VB/ < /^\(be|am|are|is|was|were|been|'m|'s|'re$/) ) < (VP < VBN << (@NP < ( -NONE- < /^\*-([0-9]+)#1%i)))))
Today's Topics

1. Passives in the WSJ PTB
2. MXPOST
3. BIKEL COLLINS
4. EVALB
MXPOST

• A statistical POS tagger:

A maximum entropy model for Part-of-Speech Tagging (Rathnaparkhi, A.)
www.aclweb.org/anthology/W96-0213

a statistical model which trains from a corpus annotated with Part-Of-Speech tags and assigns them to previously unseen text with state-of-the-art accuracy (96.6%). The model can be classified as a Maximum Entropy model and simultaneously uses many contextual “features” to predict the POS tag.

or more likely, that any corpus based algorithm on the Penn Treebank Wall St. Journal corpus will not perform much higher than 96.5% due to consistency problems.

• Find, download and unpack jmx.tar
• CLASSPATH environment variable:
  – export CLASSPATH=/Users/sandiway/Downloads/jmx/mxpost.jar
• Test file (~/Desktop/test.txt):
  – Colorless green ideas sleep furiously.

**MXPOST**

Run *(assuming I'm in my jmx directory)*:
- `.mxpost tagger.project/ < ~/Desktop/test.txt`

Read 11692 items from `tagger.project//word.voc`
Read 45 items from `tagger.project//tag.voc`
Read 42680 items from `tagger.project//tagfeatures-contexts`
Read 42680 contexts, 117558 numFeatures from `tagger.project//tagfeatures.fmap`
Read model `tagger.project//model` : numPredictions=45,
  numParams=117558
Read tagdict from `tagger.project//tagdict`

*This is MXPOST (Version 1.0)*

*Copyright (c) 1997 Adwait Ratnaparkhi*

Colorless_JJ green_JJ ideas_NNS sleep_VBP furiously_RB .
Sentence: 0 Length: 6 Elapsed Time: 0.011 seconds.
Task: to determine $t_i$ for $w_i$

Training: use features based on history:

\[- \{ w_i, w_{i+1}, w_{i-1}, w_{i+2}, w_{i-2}, t_{i-1}, t_{i-2} \} \]

<table>
<thead>
<tr>
<th>Condition</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_i$ is not rare</td>
<td>$w_i = X$ &amp; $t_i = T$</td>
</tr>
<tr>
<td>$w_i$ is rare</td>
<td>$X$ is prefix of $w_i$, $</td>
</tr>
<tr>
<td></td>
<td>$X$ is suffix of $w_i$, $</td>
</tr>
<tr>
<td></td>
<td>$w_i$ contains number &amp; $t_i = T$</td>
</tr>
<tr>
<td></td>
<td>$w_i$ contains uppercase character &amp; $t_i = T$</td>
</tr>
<tr>
<td></td>
<td>$w_i$ contains hyphen &amp; $t_i = T$</td>
</tr>
<tr>
<td>$\forall w_i$</td>
<td>$t_{i-1} = X$ &amp; $t_i = T$</td>
</tr>
<tr>
<td></td>
<td>$t_{i-2}t_{i-1} = XY$ &amp; $t_i = T$</td>
</tr>
<tr>
<td></td>
<td>$w_{i-1} = X$ &amp; $t_i = T$</td>
</tr>
<tr>
<td></td>
<td>$w_{i-2} = X$ &amp; $t_i = T$</td>
</tr>
<tr>
<td></td>
<td>$w_{i+1} = X$ &amp; $t_i = T$</td>
</tr>
<tr>
<td></td>
<td>$w_{i+2} = X$ &amp; $t_i = T$</td>
</tr>
</tbody>
</table>

1. "rare" = (count < 5)
2. feature must occur 10 times or more to be used

Table 1: Features on the current history $h_i$
MXPOST

• Example:

<table>
<thead>
<tr>
<th>Word:</th>
<th>the</th>
<th>stories</th>
<th>about</th>
<th>well-heeled</th>
<th>communities</th>
<th>and</th>
<th>developers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag:</td>
<td>DT</td>
<td>NNS</td>
<td>IN</td>
<td>JJ</td>
<td>NNS</td>
<td>CC</td>
<td>NNS</td>
</tr>
<tr>
<td>Position:</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 2: Sample Data

\[
\begin{align*}
w_i &= \text{about} & & & t_i &= \text{IN} \\
w_{i-1} &= \text{stories} & & & t_i &= \text{IN} \\
w_{i-2} &= \text{the} & & & t_i &= \text{IN} \\
w_{i+1} &= \text{well-heeled} & & & t_i &= \text{IN} \\
w_{i+2} &= \text{communities} & & & t_i &= \text{IN} \\
t_{i-1} &= \text{NNS} & & & t_i &= \text{IN} \\
t_{i-2}t_{i-1} &= \text{DT NNS} & & & t_i &= \text{IN}
\end{align*}
\]

Table 3: Features Generated From $h_3$ (for tagging about) from Table 2
**MXPOST**

- Example:

<table>
<thead>
<tr>
<th>Word:</th>
<th>the stories about well-heeled communities and developers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag:</td>
<td>DT  NNS   IN   JJ   NNS  CC  NNS</td>
</tr>
<tr>
<td>Position:</td>
<td>1  2  3  4  5  6  7</td>
</tr>
</tbody>
</table>

- \( w_{i-1} = \text{about} \) & \( t_i = \text{JJ} \)
- \( w_{i-2} = \text{stories} \) & \( t_i = \text{JJ} \)
- \( w_{i+1} = \text{communities} \) & \( t_i = \text{JJ} \)
- \( w_{i+2} = \text{and} \) & \( t_i = \text{JJ} \)
- \( t_{i-1} = \text{IN} \) & \( t_i = \text{JJ} \)
- \( t_{i-2} t_{i-1} = \text{NNS IN} \) & \( t_i = \text{JJ} \)
- prefix(\( w_i \))=w & \( t_i = \text{JJ} \)
- prefix(\( w_i \))=we & \( t_i = \text{JJ} \)
- prefix(\( w_i \))=wel & \( t_i = \text{JJ} \)
- prefix(\( w_i \))=well & \( t_i = \text{JJ} \)
- suffix(\( w_i \))=d & \( t_i = \text{JJ} \)
- suffix(\( w_i \))=ed & \( t_i = \text{JJ} \)
- suffix(\( w_i \))=led & \( t_i = \text{JJ} \)
- suffix(\( w_i \))=eled & \( t_i = \text{JJ} \)
- \( w_i \) contains hyphen & \( t_i = \text{JJ} \)

*Table 4: Features Generated From \( h_4 \) (for tagging well-heeled) from Table 2*
Another model (I downloaded) trained on wsj-02-21:

./mxpost ~/research/jmx/wsj-02-21.mxpost/ < ~/Desktop/test.txt
Read 1061 items from /Users/sandiway/research/jmx/wsj-02-21.mxpost//word.voc
Read 45 items from /Users/sandiway/research/jmx/wsj-02-21.mxpost//tag.voc
Read 4190 items from /Users/sandiway/research/jmx/wsj-02-21.mxpost//tagfeaturescontexts
Read 4190 contexts, 9286 numFeatures from /Users/sandiway/research/jmx/wsj-02-21.mxpost//tagfeatures.fmap
Read model /Users/sandiway/research/jmx/wsj-02-21.mxpost//model : numPredictions=45, numParams=9286
Read tagdict from /Users/sandiway/research/jmx/wsj-02-21.mxpost//tagdict
*This is MXPOST (Version 1.0)*
*Copyright (c) 1997 Adwait Ratnaparkhi*

Colorless JJ green JJ ideas NNS sleep IN furiously RB .
Sentence: 0 Length: 6 Elapsed Time: 0.012 seconds.
Today's Topics

1. Passives in the WSJ PTB
2. MXPOST
3. BIKEI COLLINS
4. EVALB
Bikel Collins

• Parsing
  – Command

    parse <max. heap> <settings> <derived data file> \ 
    <input file>

  – Input file format (sentences)

    The input file should have one of two Lisp-style formats:

    1. (((word1 (pos1)) (word2 (pos2)) ... (wordN (posN))))

    2. (word1 word2 ... wordN)

    Here is the first sentence of Section 00 of the WSJ Penn Treebank in Format 1:

    ((Pierre (NNP)) (Vinken (NNP)) (; (;) (;)) (61 (CD))
    (years (NNS)) (old (JJ)) (; (;) (;) (will (MD)) (join (VB))
    (the (DT)) (board (NN)) (as (IN)) (a (DT))
    (nonexecutive (JJ)) (director (NN)) (Nov. (NNP))
    (29 (CD)) (. (.))))
Statistical Parser Development

• **Methodology**
  – partition corpus into a training and a test set
  – compare parser output on test set against withheld reference parses

• **Possible criteria**
  – **all or nothing**: match the reference ("gold") tree exactly
  – **partial credit**: use some "goodness of fit" metric
Statistical Parser Development

- Standard Operating Procedure:
  - one million words of 1989 Wall Street Journal (WSJ) articles
  - nearly 50,000 sentences (49,208), sectioned
Computing Tree Similarity

Evaluation:

– we’ve got 2400 sentences to compare

• How do we **automate** parser output scoring?

**PARSEVAL:** bracketing span match


**Computer program:** EVALB (Sekine & Collins, 1997)
Today's Topics

1. Passives in the WSJ PTB
2. MXPOST
3. BIKEEL COLLINS
4. EVALB
Proposal: given trees $T_1$ and $T_2$

- **Preprocessing stage:**
  - Remove auxiliary verbs, *not*, infinitival marker *to*, empty categories, possessive ‘s, and punctuation
  - Remove unary branching nodes (including part of speech tags)

- **Compute scores:**
  - # Crossing Parentheses
  - Recall
  - Precision

In the literature, the F-score is reported, i.e.

$$F = 2 \cdot \text{precision} \cdot \text{recall} \div (\text{precision} + \text{recall})$$
PARSEVAL

Proposal: given trees $T_1$ and $T_2$

• Preprocessing stage:
  – Remove auxiliary verbs, *not*, infinitival marker *to*,
    empty categories, possessive ‘s, and punctuation
  – Remove unary branching nodes (including part of
    speech tags)

<table>
<thead>
<tr>
<th>Original</th>
<th>Simplified</th>
</tr>
</thead>
<tbody>
<tr>
<td>would  go there</td>
<td>go there</td>
</tr>
<tr>
<td>has been laughing</td>
<td>laughing</td>
</tr>
<tr>
<td>does  sing it correctly</td>
<td>sing it correctly</td>
</tr>
<tr>
<td>is not in here</td>
<td>is in here</td>
</tr>
<tr>
<td>she opted to retire</td>
<td>she opted retire</td>
</tr>
<tr>
<td>Lori’s mother</td>
<td>Lori mother</td>
</tr>
<tr>
<td>is a cup</td>
<td>(not deleted if copula or main verb)</td>
</tr>
<tr>
<td>is blue</td>
<td></td>
</tr>
<tr>
<td>has  a dollar</td>
<td></td>
</tr>
<tr>
<td>does  the laundry</td>
<td></td>
</tr>
</tbody>
</table>
Proposal: given trees $T_1$ and $T_2$

- Preprocessing stage:
  - Remove auxiliary verbs, *not*, infinitival marker *to*, empty categories, possessive ‘s, and punctuation
  - Remove unary branching nodes (including part of speech tags)
PARSEVAL

Proposal: given trees $T_1$ and $T_2$

• Preprocessing stage:
  – Remove auxiliary verbs, *not*, infinitival marker *to*, empty categories, possessive ‘*s’, and punctuation
  – Remove unary branching nodes (including part of speech tags)

Black et al. (1991) describes other special case rules for sequences of prepositions etc.
PARSEVAL

Proposal: given trees $T_1$ and $T_2$

- Compute scores:
  - # Crossing Parentheses
  - Recall
  - Precision
Proposal: given trees $T_1$ and $T_2$

- Compute scores:
  - $\#$ Crossing Parentheses

Score $= 1$
The prospect of cutting back spending is easy to compute.

**PARSEVAL**

### # Crossing Parentheses

0. The prospect of cutting back spending

**Gold (T2):**

<table>
<thead>
<tr>
<th>Constituent</th>
<th>Span</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-6</td>
</tr>
<tr>
<td>2</td>
<td>0-2</td>
</tr>
<tr>
<td>3</td>
<td>2-6</td>
</tr>
<tr>
<td>4</td>
<td>3-6</td>
</tr>
</tbody>
</table>

**Parser (T1):**

<table>
<thead>
<tr>
<th>Constituent</th>
<th>Span</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-6</td>
</tr>
<tr>
<td>2</td>
<td>1-6</td>
</tr>
<tr>
<td>3</td>
<td>2-6</td>
</tr>
<tr>
<td>4</td>
<td>3-6</td>
</tr>
<tr>
<td>5</td>
<td>3-5</td>
</tr>
</tbody>
</table>

The prospect of cutting back spending is easy to compute.
PARSEVAL

Proposal: given trees $T_1$ and $T_2$ (gold)

- Compute scores:
  - # Crossing Parentheses
  - Recall
  - Precision
PARSEVAL

Proposal: given trees $T_1$ and $T_2$ (gold)

- Compute scores:
  - Recall
  $\frac{|\text{nodes}(T_1) \cap \text{nodes}(T_2)|}{|\text{nodes}(T_2)|}$
  - Precision
  $\frac{|\text{nodes}(T_1) \cap \text{nodes}(T_2)|}{|\text{nodes}(T_1)|}$
Proposal: given trees $T_1$ and $T_2$ (gold)

- Compute scores:
  - Recall: $\frac{\text{nodes}(T_1) \cap \text{nodes}(T_2)}{\text{nodes}(T_2)}$
  - Precision

Gold ($T_2$):

- Recall: $\frac{3}{4} = 75\%$
- Precision: $\frac{3}{5} = 60\%$

F-score: $2 \cdot \frac{3}{5} \cdot \frac{3}{4} \div (\frac{3}{5} + \frac{3}{4}) = 69\%$
Recall and Precision:

\[
\text{The prospect of cutting back spending}
\]

**PARSEVAL**

**Parser (T₁):**

```
1. The 
2. prospect 
3. of 
4. cutting 
5. back 
6. spending 
```

**Gold (T₂):**

```
1. also easy to compute 
2. The 
3. prospect 
4. of 
5. cutting 
6. back 
    7. spending 
```

Constituent Span

- 1: 0 - 6
- 2: 0 - 2
- 3: 2 - 6
- 4: 3 - 6
- 1: 0 - 6
- 2: 1 - 6
- 3: 2 - 6
- 4: 3 - 6
- 5: 3 - 5

Also easy to compute
EVALB

http://nlp.cs.nyu.edu/evalb/

Implements the basic PARSEVAL proposal
(some small differences)

• Preprocessing stage

```
DELETE_LABEL TOP
DELETE_LABEL -NONE-
DELETE_LABEL ,
DELETE_LABEL :
DELETE_LABEL ``
DELETE_LABEL `'`
DELETE_LABEL .
```
```
## Equivalent labels, with ## unlabeled or labeled bracketing
## the pairs are counted: ## 0: unlabeled bracketing
## This is non-directional. ## 1: labeled bracketing
```

```
# EQ_LABEL ADVP PRT
```

```
# EQ_WORD  Example example
```

• Compute scores:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>40</td>
<td>0</td>
<td>67.74</td>
<td>72.41</td>
<td>21</td>
<td>31</td>
<td>29</td>
<td>7</td>
<td>37</td>
</tr>
<tr>
<td>3</td>
<td>31</td>
<td>0</td>
<td>71.43</td>
<td>78.95</td>
<td>15</td>
<td>21</td>
<td>19</td>
<td>2</td>
<td>26</td>
</tr>
<tr>
<td>4</td>
<td>35</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>0</td>
<td>32</td>
</tr>
</tbody>
</table>
[6] THE PARAMETER (.prm) FILE

The .prm file sets options regarding the scoring method. COLLINS.prm gives the same scoring behaviour as the scorer used in (Collins 97). The options chosen were:

1) Labeled 1
   to give labelled precision/recall figures, i.e. a constituent must have the same span *and* label as a constituent in the gold file.

2) DELETE_LABEL TOP
   Don’t count the "TOP" label (which is always given in the output of tgrep) when scoring.

3) DELETE_LABEL -NONE-
   Remove traces (and all constituents which dominate nothing but traces) when scoring. For example
   .... (VP (VBD reported) (SBAR (-NONE- 0) (S (-NONE- *T*-1))))
   (..))
   would be processed to give
   .... (VP (VBD reported)) (..))

4) DELETE_LABEL ,  -- for the purposes of scoring remove punctuation
   DELETE_LABEL :
   DELETE_LABEL "
   DELETE_LABEL "
   DELETE_LABEL .

5) DELETE_LABEL_FOR_LENGTH -NONE-  -- don’t include traces when calculating the length of a sentence (important when classifying a sentence as <=40 words or >40 words)

6) EQ_LABEL ADVP PRT
   Count ADVP and PRT as being the same label when scoring.
EVALB

• To run the scorer:
• > evalb -p Parameter_file Gold_file Test_file

• For example to use the sample files:
• > evalb -p sample.prm sample.gld sample.tst
### Gold standard:

- \( (S \ A \ (P \ this)) \ (B \ (Q \ is) \ (A \ (R a) \ (T \ test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)

### Test:

- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)
- \( (S \ A \ (P \ this)) \ (B \ (Q is) \ (A \ (R a) \ (T test))) \)

---

**EVALB**
### EVALB

#### Results:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>0</td>
<td>75.00</td>
<td>75.00</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>75.00</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0</td>
<td>75.00</td>
<td>75.00</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>75.00</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>0</td>
<td>75.00</td>
<td>75.00</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>0</td>
<td>50.00</td>
<td>66.67</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>0</td>
<td>25.00</td>
<td>100.00</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>0</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>80.00</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>50.00</td>
<td>4</td>
<td>4</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>11</td>
<td>4</td>
<td>2</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0.00</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0.00</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0.00</td>
</tr>
<tr>
<td>14</td>
<td>4</td>
<td>2</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0.00</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0.00</td>
</tr>
<tr>
<td>17</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>18</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>19</td>
<td>4</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0.00</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>21</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>34</td>
<td>34</td>
<td>34</td>
<td>34</td>
<td>44</td>
<td>44</td>
<td>100.00</td>
</tr>
<tr>
<td>22</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>23</td>
<td>4</td>
<td>0</td>
<td>100.00</td>
<td>100.00</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>100.00</td>
</tr>
<tr>
<td>24</td>
<td>4</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0.00</td>
</tr>
</tbody>
</table>

--- Summary ---

--- All ---

<table>
<thead>
<tr>
<th>Number of sentence</th>
<th>Number of Error sentence</th>
<th>Number of Skip sentence</th>
<th>Number of Valid sentence</th>
<th>Bracketing Recall</th>
<th>Bracketing Precision</th>
<th>Complete match</th>
<th>Average crossing</th>
<th>No crossing</th>
<th>2 or less crossing</th>
<th>Tagging accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>5</td>
<td>2</td>
<td>17</td>
<td>87.76</td>
<td>90.53</td>
<td>52.94</td>
<td>0.06</td>
<td>94.12</td>
<td>100.00</td>
<td>98.15</td>
</tr>
</tbody>
</table>

--- len<=40 ---

<table>
<thead>
<tr>
<th>Number of sentence</th>
<th>Number of Error sentence</th>
<th>Number of Skip sentence</th>
<th>Number of Valid sentence</th>
<th>Bracketing Recall</th>
<th>Bracketing Precision</th>
<th>Complete match</th>
<th>Average crossing</th>
<th>No crossing</th>
<th>2 or less crossing</th>
<th>Tagging accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>5</td>
<td>2</td>
<td>16</td>
<td>81.25</td>
<td>85.25</td>
<td>50.00</td>
<td>0.06</td>
<td>93.75</td>
<td>100.00</td>
<td>96.88</td>
</tr>
</tbody>
</table>
[5] HOW TO CREATE A GOLDFILE FROM THE PENN TREEBANK

The gold and parsed files are in a format similar to this:

(TOP (S (INTJ (RB No)) (, ,) (NP (PRP it)) (VP (VBD was) (RB n't) (NP (NNP Black) (NNP Monday))) (, .)))

To create a gold file from the treebank:

tgrep -wn '/.*/' | tgrep_proc.prl

will produce a goldfile in the required format. ("tgrep -wn '/.*/'" prints parse trees, "tgrep_process.prl" just skips blank lines).

For example, to produce a goldfile for section 23 of the treebank:

tgrep -wn '/.*/' | tail +90895 | tgrep_process.prl | sed 2416q > sec23.gold
• However you can use tсужer on from the Stanford tregex you downloaded to accomplish the same thing

• Example:
  – file: wsj_0927.mrg
EVALB

./tsurgeon.sh -treeFile wsj_0927.mrg -s

( (S (NP-SBJ-1 (NNP H.) (NNP Marshall) (NNP Schwarz)) (VP (VBD was) (VP (VBN named) (S (NP-SBJ (-NONE- *-1)) (NP-PRD (NP (NN (NN chairman)) (CC and) (NP (NN (NN chief) (JJ executive) (NN officer))) (PP (IN of) (NP (NP (NNP U.S.) (NNP Trust) (NNP Corp.))) (,) (NP (NP (DT a) (JJ private-banking) (NN firm)) (PP (IN with) (NP (NP (NNS assets)) (PP (IN under) (NP (NN management))) (PP (IN of) (NP (QP (IN about) ($) $) (CD 17) (CD billion)) (-NONE- *U*)))))))) (.) (.)

( (S (NP-SBJ-1 (NP (NNP Mr.) (NNP Schwarz))) (,) (ADJP (NP (CD 52) (NNS years)) (JJ old)) (,) (VP (MD will) (VP (VB succeed) (NP (NP (NNP Daniel) (NNP Davison)) (NP-TMP (NNP Feb.) (CD 1)) (,) (SBAR-TMP (RB soon) (IN after) (S (NP-SBJ (NP (NNP Mr.) (NNP Davison))) (VP (VBZ reaches) (NP (NP (NP (DT the) (NN company) (POS 's)) (JJ mandatory) (NN retirement) (NN age)) (PP (IN of) (NP (NP (CD 65)))))))))) (,) (.)

( (S (NP-SBJ-1 (NP (NNP Mr.) (NNP Schwarz))) (,) (SBAR (WHNP-2 (WP who)) (S (NP-SBJ (-NONE- *T*-2)) (VP (VBZ is) (NP-PRD (NP (NN (NN president)) (PP (IN of) (NP (NP (NNP U.S.) (NNP Trust))) (,) (,) (VP (MD will) (VP (VB be) (VP (VBN succeeded) (NP (-NONE- *-1)) (PP-LOC (IN in) (NP (DT that) (NN post))) (PP (IN by) (NP-LGS (NP (NP (NN Jeffrey) (NNP S.) (NNP Maurer)) (,) (NP (CD 42)) (,) (SBAR (WHNP-3 (WP who)) (S (NP-SBJ (-NONE- *T*-3)) (VP (VBZ is) (NP-PRD (NP (JJ executive) (NN vice (NN president))) (PP (IN in) (NP (NP (NN charge))) (PP (IN of) (NP (NP (DT the) (NN company) (POS 's)) (JJ mandatory) (NN asset-management) (NN group))))))) (,) (.)

( (S (NP-SBJ (NP (NNP U.S.) (NNP Trust))) (,) (NP (NP (DT a) (JJ 136-year-old) (NN institution)) (SBAR (WHNP-2 (WDT that)) (S (NP-SBJ (-NONE- *T*-2)) (VP (VBZ is) (NP-PRD (NP (CD one)) (PP (IN of) (NP (NP (DT the) (JJS earliest) (NN high-net) (JJ worth) (NNS banks)) (PP-LOC (IN in) (NP (DT the) (NNP U.S.))))))) (,) (,) (VP (VBZ has) (VP (VBN faced) (NP (NP (VBG intensifying) (NN competition))) (PP (IN from) (NP (NP (JJS other) (NNS firms)) (SBAR (WHNP-3 (WDT that)) (S (NP-SBJ (-NONE- *T*-3)) (VP (VBP have) (VP (VP (VBN established) (NP (-NONE- *RNR*-1)) (,) (CC and) (VP (ADVP-MNR (RB heavily)) (VBN promoted) (NP (-NONE- *RNR*-1))) (,) (NP-1 (NP (JJ private-banking) (NNS businesses)) (PP (IN of) (NP (PRP$ their) (JJ own))))))) (,) (.)

• You can then redirect standard output to a file ...
EVALB

Example

• Assemble section 23 into one file
  
cat TREEBANK_3/parsed/mrg/wsj/23/*.mrg >
  
wsj_23.mrg
  
– at this point not one tree per physical line

• Run tsurgeon
  
./tsurgeon.sh -treeFile wsj_23.mrg -s > wsj_23.gold
  
– File wsj_23.gold contains one tree per line
Task

• Run section 23 of the WSJ Treebank on the Bikel Collins parser
  – Extract the sentences from section 23 (perl etc.)
• Then run EVALB on the section to see how the Bikel Collins parser scores. Report back next time.

1 No, it was n't Black Monday .
2 But while the New York Stock Exchange did n't fall apart Friday as the Dow Jones Industrial Average plunged 190.58 points -- most of it in the final hour -- it barely managed to stay this side of chaos .