Introduction to Computational Linguistics: Review Lecture

Adam Meyers
New York University
Summary

• Basic Instructions for the Final Exam
• Regular Expressions and Phrase Structure Rules
  – Writing them and applying them to data
• Algorithms for classifying words & phrases
  – Viterbi, HMM, parsing
• Annotating phrase structure and POS
• Reference Resolution
  – Binding Theory, Hobbs Search
• Calculating Evaluation Measures
• Feature Structure (quick example)
• Machine Translation
  – EM and Decoding
• Following Annotation Guidelines/Doing Linguistic Analyses
• Additional Questions
Final Exam

• Open book, Open notes, calculator is OK
• You have approximately 1 hour and 50 minutes to do it – it is OK to leave early if you are done.
• You should put your name on all test materials.
• It should be easy for me to find your answers. If you put them anywhere, but on the test itself, please include a note so that I can find it.
• The sample test is an approximation of the real test
  – There may be stuff on the real test that is not on the practice and vice versa
  – The real test will be no longer than the practice test – it may be shorter
• Strategy: Do all the fast questions first
  – Initially, do not spend more than 7 minutes on a question.
  – Then go back and complete what you didn't on the first round
Regexp = formula specifying set of strings

- Regexp = \( \emptyset \)
  - The empty set
- Regexp = \( \varepsilon \)
  - The empty string
- Regexp = a sequence of one or more characters from the set of characters
  - \( X \)
  - \( Y \)
  - *This sentence contains characters like &T^**%P*
- Disjunctions, concatenation, and repetition of regexps yield new regexps
Concatenation, Disjunction, Repetition

• Concatenation
  – If $X$ is a regexp and $Y$ is a regexp, then $XY$ is a regexp
  – Examples
    • If $ABC$ and $DEF$ are regexps, then $ABCDEF$ is a regexp
    • If $AB^*$ and $BC^*$ are regexps, then $AB^*BC^*$ is a regexp
      – Note: Kleene * is explained below

• Disjunction
  – If $X$ is a regexp and $Y$ is a regexp, then $X | Y$ is a regexp
  – Example: $ABC | DEF$ will match either $ABC$ or $DEF$

• Repetition
  – If $X$ is a regexp than a repetition of $X$ will also be a regexp
    • The Kleene Star: $A^*$ means 0 or more instances of $A$
    • Regexp{number}: $A^2$ means exactly 2 instances of $A$
• Disjunction of characters
  – \([ABC]\) – means the same thing as \(A \mid B \mid C\)
  – \([a-zA-Z0-9]\) – ranges of characters equivalent to listing characters, e.g., \(a\mid b\mid c\ldots\mid A\mid B\ldots\mid 0\mid 1\ldots\mid 9\)
  – ^ inside of bracket means complement of disjunction, e.g., \([^a-z]\) means a character that is neither \(a\) nor \(b\) nor \(c\) … nor \(z\)

• Parentheses
  – Disambiguate scope of operators
    • \(A(BC)\mid(DEF)\) means \(ABC\) or \(ADEF\)
    • Otherwise defaults apply, e.g., \(ABC\mid D\) means \(ABC\) or \(ABD\)

• ? signifies optionality
  – \(ABC?\) is equivalent to \((ABC)\mid(AB)\)

• + indicates 1 or more
  – \(A(BC)^*\) is equivalent to \(A\mid(\text{A}(BC)^+)\)
Regexp Notation Slide 3

• Special Symbols:
  – $A.*B$ – matches A and B and any characters between (period = any character)
  – $^ABC$ – matches ABC at beginning of line (^ represents beginning of line)
  – $[.?!]$ – matches sentence final punctuation ($$ represents end of line)
• Python's Regexp Module
  – Searching
    • Groups and Group Numbers
  – Compiling
  – Substitution
• Similar Modules for: Java, Perl, etc.
A Sample Regular Expression: In Class

- A regular expression for a date that will include the following expressions:
  - January 3, 2012
  - January, 2012
  - January 3
  - Jan. 3, 2012
- Specific Constraints:
  - Allow all the months of the year, as well as abbreviations that consist of the first 3 letters of the month and a period
  - The day should be a one or two digit number
  - The year should be a four digit number
  - The day and year are optional
  - A comma and a space precedes the year.
Chomsky Hierarchy

- Type 0: Anything $\rightarrow$ Anything
- Type 1: Context-Sensitive Rules: $\alpha A \beta \rightarrow \alpha \gamma \beta$
  - For example,
    - DUCK DUCK DUCK $\rightarrow$ DUCK DUCK GOOSE
    - Means convert DUCK to a GOOSE, if preceded by 2 DUCKS
- Type 2: Context-free rules: $A \rightarrow \alpha \gamma \beta$
  - Left-hand side is always one nonterminal
  - Type of rules used for most natural language parsing
- Type 3: Regular (finite state) grammars
  - $A \rightarrow \beta a$ or $A \rightarrow \epsilon$ (left regular)
  - $A \rightarrow a \beta$, or $A \rightarrow \epsilon$ (right regular)
  - Regular expressions are type 3 rules
- $Type_0 \supseteq Type_1 \supseteq Type_2 \supseteq Type_3$
Phrase Structure Rules and Tree

• Lets mark Parts of Speech, Draw a Phrase Structure Tree, and List the Rules for the following sentence from Wikipedia:
  – *Parodia tenuicylindrica* is a small species of cactus native to the Rio Grande do Sul region of Brazil
  – We will assume that:
    • species names consist of proper nouns
    • all the words in proper noun phrases are proper nouns
  – This sentence has other things about it that make it slightly more difficult than I would choose for the test
Chomsky Normal Form (required by CKY)

- Context Free Grammars can be converted to CNN
  - 3 types of rules:
    - $XP \rightarrow YP ZP$  Binary Branching
    - $XP \rightarrow x$       NonTerminal $\rightarrow$ terminal
    - $XP \rightarrow \varepsilon$  NonTerminal $\rightarrow$ empty string

- Conversion
  - Replace $VP \rightarrow VG \; \& \; NP \rightarrow NG$ with nonbranching rules expanding VP and NP to whatever VG and NG matched to:
    - **Not good examples:** $VP \rightarrow ate$, $VP \rightarrow had$, $NP \rightarrow food$, …
    - Usually V and N are assumed to be nonterminals, and the POS of words are looked up
  - Replace $VP \rightarrow V NP PP$ with 2 rules:
    - $VP \rightarrow VG PP$
    - $VG \rightarrow V NP$
  - Replace: $NP \rightarrow POSSP N PP$ with 2 rules:
    - $NP \rightarrow NG PP$
    - $NG \rightarrow POSSP N$
CKY Recognizer and Parser

- Create a (triangular) table representing all spans in the sentence from 0 (the position before the first word) to N the position after sentence of length N
- For j from 1 to N do:
  - Fill in one span of length 1 using a POS rules, e.g., V → ate
  - On different iterations these will be [0, 1], [1, 2], …, [N-1, N]
  - For i from 0 to j-2 do: ### Note: J&M does this in reverse
    - for k from i+1 to j-1:
      - Add all matching nonterminals to [i,j] in table * ## fill in rest of column j
- * A nonterminal matches iff
  - There is some rule of the form A--> BC in the grammar
  - [i,k] includes the label B and [k,j] includes the label C
- Note that the inner 2 loops identify all [i,k] amd [k,j] such that i is between 0 and j-2, i<k and k<j, thus identifying the possible binary partitions of [0,j]
- Parser needs 2 additional things:
  - Differentiate different expansions of same nonterminal
  - Record pointers to matched children of nonterminals
Recognize with CKY & Grammar'
Outer Loop 1\textsuperscript{st} Iteration

<table>
<thead>
<tr>
<th>The</th>
<th>clam</th>
<th>'s</th>
<th>group</th>
<th>had</th>
<th>knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>0</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[0,1]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Computational Linguistics
Review Lecture
2011-2012
### 2\textsuperscript{nd} Iteration

<table>
<thead>
<tr>
<th></th>
<th>The</th>
<th>clam</th>
<th>'s</th>
<th>group</th>
<th>had</th>
<th>knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>D</td>
<td>NP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>[0,1]</td>
<td>[0,2]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 3rd Iteration

<table>
<thead>
<tr>
<th></th>
<th>The</th>
<th>clam</th>
<th>'s</th>
<th>group</th>
<th>had</th>
<th>knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>D [0,1]</td>
<td>NP [0,2]</td>
<td>POSSP [0,3]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>N, NP [1,2]</td>
<td>POSSP [1,3]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The clam's group had knowledge.
# 4th Iteration

<table>
<thead>
<tr>
<th></th>
<th>The</th>
<th>clam</th>
<th>'s</th>
<th>group</th>
<th>had</th>
<th>knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>D [0,1]</td>
<td>NP [0,2]</td>
<td>POSSP [0,3]</td>
<td>NP [0,4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>N, NP [1,2]</td>
<td>POSSP [1,3]</td>
<td>NP [1,4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>POSS [2,3]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>N, NP [3,4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Computational Linguistics  
Review Lecture  
2011-2012
### 5th Iteration

<table>
<thead>
<tr>
<th></th>
<th>The</th>
<th>clam</th>
<th>'s</th>
<th>group</th>
<th>had</th>
<th>knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>D [0,1]</td>
<td>NP [0,2]</td>
<td>POSSP [0,3]</td>
<td>NP [0,4]</td>
<td>S [0,5]</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>N, NP [1,2]</td>
<td>POSSP [1,3]</td>
<td>NP [1,4]</td>
<td>S [1,5]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>POSS [2,3]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>N,NP [3,4]</td>
<td>S [3,5]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>V, VP [4,5]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The clam's group had knowledge.
<table>
<thead>
<tr>
<th>The</th>
<th>clam</th>
<th>'s</th>
<th>group</th>
<th>had</th>
<th>knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>0</td>
<td>D [0,1]</td>
<td>NP [0,2]</td>
<td>POSSP [0,3]</td>
<td>NP [0,4]</td>
<td>S [0,5]</td>
</tr>
<tr>
<td>1</td>
<td>N, NP [1,2]</td>
<td>POSSP [1,3]</td>
<td>NP [1,4]</td>
<td>S [1,5]</td>
<td>S [1,6]</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Earley Parse Trace: 1

- Sentence with position numbers: 0 John 1 's 2 book 3 fell 4
- Rules
  - $S \rightarrow NP \ VP$  \ (Assume $S$ is the only initial symbol)
  - $NP \rightarrow N$
  - $NP \rightarrow PossP \ N$
  - $PossP \rightarrow NP \ 's$
  - $VP \rightarrow V$
- 1. Initialize: Put $S \rightarrow NP \ VP \ [0,0]$ in chart
- 2. Pred: Put $NP \rightarrow PossP \ N \ [0,0]$ in chart
- 3. Pred: Put $NP \rightarrow N \ [0,0]$ in chart
- 4. Pred: Put $PossP \rightarrow NP \ 's \ [0,0]$ in chart
  - Do not add $NP \rightarrow PossP \ N \ [0,0]$ in chart a 2\textsuperscript{nd} time
- 5. Scan and put $NP \rightarrow John . \ [0,1]$ in chart
Earley Parse Trace: 2

- 6. Completer adds $S \rightarrow NP. VP$ [0,1]
- 7. Predictor adds $VP \rightarrow . V$ [1,1]
- 8. Completer adds $PossP \rightarrow NP. 's$ [0,1]
- 9. Scanner adds $PossP \rightarrow NP 's .$ [0,2]
- 10. Completer adds $NP \rightarrow PossP. N$ [0,2]
- 11. Scanner adds $NP \rightarrow PossP. book$ [0,3]
- 12. Completer adds $S \rightarrow NP. VP$ [0,3]
- 13. Predictor adds $VP \rightarrow . V$ [3,3]
- 14. Scanner adds $VP \rightarrow fell.$ [3,4]
- 15. Completer adds $S \rightarrow NP VP.$ [0,4]

- Complete parse of sentence found
Viterbi Decoding of HMM for *rose pickles*

- **Likelihood:**
  - *rose*: NNP .01, NN .02, VBD .05
  - *pickles*: NNP .001, NNS .03, VBZ .05

- **Transition Probabilities:**
**Rose Pickles**

- **Likelihood:**
  - *rose*: NNP .01, NN .02, VBD .05
  - *pickles*: NNP .001, NNS .03, VBZ .05
- **Fill in: max (previous X transition X likelihood)**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1: Rose</th>
<th>2: Pickles</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>NNP</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>NNS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>NN</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>VBZ</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>VBD</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>End</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Binding Theory for English 3rd Pers Prons

• Case 1: If the pronoun *p* is inside an NP premodified by a possessive, the antecedent needs to be outside of this NP
  
  – \textit{John} likes Mary's drawing of \textit{him}
  
  – \textit{John} likes \textit{his} drawing of Mary

• Case 2: Otherwise, the antecedent must be outside the immediate tensed clause containing the personal pronoun.
  
  – \textit{John} said that \textit{he} liked pizza.
  
  – \textit{John} wanted for \textit{him} to like pizza.
  
  – \textit{John} liked \textit{him}.

• Theories of binding vary about how these (and similar) constraints are encoded, but the differences in the final result (quality of system output) is minimal. While these 2 rules cover most cases, there are also some exceptions:
  
  – \textit{John} always carries a slice of pizza with \textit{him}.
Binding Theory for English Reflexives/Reciprocals

• The antecedent of a reflexive/reciprocal must be the closest subject or possessive such that:
  – The antecedent precedes and “commands” the pronoun
    • A commands B if A is the sibling of a phrase that dominates B.
  – There is no possessive or subject for phrases in the path in the phrase structure tree between antecedent and pronoun

• Examples:
  – Mary saw herself vs. *Mary said that John would meet herself soon
  – Mary's picture of herself vs. *Mary saw John's picture of herself

• These rules covers most cases.
  – Exception: Pictures of themselves made the actors nervous.
Hobbs Search Algorithm to Find Antecedent of Anaphors

1. Go to NP immediately dominating pronoun
2. Go up to 1st dominating NP or S node. This node = X path to X = p.
3. Traverse branches below X to the left of p, left-to-right and and breadth first. Propose each NP n as an antecedent if there is an NP or S between n and X.
4a. Is X the highest S in the sentence? (Recursive)
   - Yes
   - No
4b. Search previous sentences in order from right to left. Search each tree from left to right, breadth 1st, proposing each NP as antecedent
5. From Node X, go up to the 1st NP or S. Call this node X and the path to X p.
6. If X = NP and there is no N' in p, propose X as antecedent
7. Search for antecedent in branches below X, left-to-right, breadth first. Propose each NP.
8. If X is an S node, traverse all branches of X preceding p, left-to-right, breadth first, but not going below any S or NP node found. Propose each NP.

End
Hobbs Search Example

1. Mary saw the chicken.

2. Jim said that she laughed.
Evaluation

- Recall/Precision/F-Score given answer key for a task
  \[
  \text{Recall} = \frac{|\text{Correct}|}{|\text{Answer Key}|} \quad \text{Precision} = \frac{|\text{Correct}|}{|\text{System Output}|} \quad F - \text{Score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}
  \]

- Kappa – How much better is interannotator agreement than chance?
  \[
  \text{Kappa} = \frac{\text{Percent (Actual Agreement)} - \text{Prob (Chance Agreement)}}{1 - \text{Prob (Chance Agreement)}}
  \]

- Coreference with the B-cubed algorithm
  - Precision: For each partition of Coreferential NPs in system output: average the percentage of links in the answer
  - Recall: For each partition of Coreferential NPs in the answer, average the percentage of links in the system output
  - Example:
    - 2 system entities: \(\{A_1, A_2, A_3, A_4, C_1\}\) and \(\{B_1, B_2\}\)
    - 2 answer key entity: \(\{A_1, A_2, A_3, A_4, B_1, B_2\}\) and \(\{C_1\}\)
    - Precision = \((4 \times 4/5) + 1/5 + 2) \times 1/7 = .77\)
    - Recall = \(((4 \times 2/3) + (2 \times 1/3) + 1) \times 1/7 = .62\)
Example Feature Structure Question

• What (Typed) Feature Structure would result if FS1 was unified with the value of the path $Quant$ in FS2?
  
  – Note: $orth$ is short for $orthography$ and $cat$ is short for $category$
Machine Translation Questions

• Decoding Question (like the sample test)
• Walk through part of Maximization/Estimation
  – Given a set of translation probabilities, calculate the probability of a particular alignment
  – Given a set of alignments and their probability scores, calculate the new translation probabilities
Simplified Example of EM model

• Given
  – 4 French words: *la, maison, bleu*, and *fleur*
  – 4 English words: *the, house, blue* and *flower*
  – We only allow 1 to 1 alignments

• Starting assumption
  – Each French word has a .25 chance of being translated as a given English word
Initial Alignment Probs for 3 E/F pairs

• *la maison → the house*
  – *la/the* (.25), *maison/the* (.25), *la/house* (.25), *maison/house* (.25)
  – *la/the X maison/house* = .0625
  – *maison/the X la/house* = .0625

• *la maison bleu → the blue house* (all possible alignments)
  – *la/the X maison/house X bleu/blue* = .25³ = .015625
  – *la/the X maison/blue X bleu/house* = .015625
  – *la/house X maison/the X bleu/blue* = .015625
  – *la/house X maison/blue X bleu/house* = .015625
  – *la/blue X maison/house X bleu/the* = .015625
  – *la/blue X maison/the X bleu/house* = .015625

• *La fleur → the flower*
  – *la/the X fleur/flower* = .0625
  – *fleur/the X la/flower* = .0625
Maximum Liklihood Estimates (MLE)

- For each e/f pair and for each sentence, add up the probabilities of alignments that contain that pair and regularize to 1 (initially: all prob=.25)
- Sum these scores and divide by the number of instances of f.
- Translations from X to the
  - la/the: .5 of the first set of alignments, .33 of the second set and .5 of the 3rd
    - \((.5 + .33 + .5) / 3 = .44\)
  - maison/the: .5 of the 1st + .33 of the 2nd, 0 in the 3rd
    - \((.5 + .33)/3 = .42 = .29\)
  - bleu/the: 0 in the 1st + .33 of the 2nd + 0 in the 3rd
    - \(.33/3 = .11\)
  - fleur/the: 0 in the 1st and 2nd, .5 in the 3rd
    - \(.5/3 = .17\)
- house: la/house=.42, maison/house=.42, bleu/house=.17, fleur/house=0
- blue: la/blue=.33, maison/blue=.33, bleu/blue= .33, fleur/blue=0
- flower: la/flower=.5 maison/flower=0, blue/flower=0, fleur/flower= .5
Expectation: Rescore Alignments

• **la maison → the house**
  – \( \text{la/the} (.44), \text{maisson/the} (.29), \text{la/house} (.42), \text{maisson/house} (.42) \)
  – \( \text{la/the X maison/house} = .1848 \)
  – \( \text{maison/the X la/house} = .1276 \)

• **la maison bleu → the blue house** (all possible alignments)
  – \( \text{la/the X maison/house X bleu/blue} = .06098 \)
  – \( \text{la/the X maison/blue X bleu/house} = .02468 \)
  – \( \text{la/house X maison/the X bleu/blue} = .04019 \)
  – \( \text{la/house X maison/blue X bleu/house} = .02356 \)
  – \( \text{la/blue X maison/house X bleu/the} = .045274 \)
  – \( \text{la/blue X maison/the X bleu/house} = .016269 \)

• **La fleur → the flower**
  – \( \text{la/the X fleur/flower} = .22000 \)
  – \( \text{fleur/the X la/flower} = .08500 \)
Translating sample sentence

- Input: *La maissan bleu*
- Translation probabilities (hypothetical):

<table>
<thead>
<tr>
<th>French</th>
<th>English</th>
<th>the</th>
<th>blue</th>
<th>house</th>
<th>flower</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>la</em></td>
<td>the</td>
<td>.70</td>
<td>.10</td>
<td>.15</td>
<td>.05</td>
</tr>
<tr>
<td><em>maisson</em></td>
<td>blue</td>
<td>.24</td>
<td>.26</td>
<td>.50</td>
<td>0</td>
</tr>
<tr>
<td><em>bleu</em></td>
<td>house</td>
<td>.25</td>
<td>.41</td>
<td>.22</td>
<td>.12</td>
</tr>
<tr>
<td><em>fleur</em></td>
<td>flower</td>
<td>.19</td>
<td>.17</td>
<td>.01</td>
<td>.63</td>
</tr>
</tbody>
</table>

- Unigram probabilities (count in WSJ ÷ 1 million)
  - the = 0.035, blue = $1.3 \times 10^{-4}$, house = $6.7 \times 10^{-4}$, flower = $6 \times 10^{-6}$
- The most probable translation would be:
  - *the house blue* = translation-prob X language prob = $4.37 \times 10^{-10}$
    - translation-prob = $0.7 \times 0.5 \times 0.41 = 0.1435$
    - Lang-prob = $0.035 \times 6.7 \times 10^{-4} \times 1.3 \times 10^{-4} = 3.05 \times 10^{-9}$
Sample SRL Annotation Task

• Mark commas with a label indicating a function as indicated:
  – **APPOSITION**: if the comma joins to NPs that are in an is-a relation, e.g., *Barack Obama, president of the U.S.*
  – **AFFILIATED**: if the comma joins to NPs that are related such that the second provides an affiliation or a larger entity that the first is a part of, e.g., a) *Adam Meyers, NYU*; b) *Columbus, Ohio*
  – **CONJUNCTION**: if the comma is between constituents that are conjoined together, e.g., *big cats, little cats, cats with fangs, or regular cats*
  – **ADVERBIAL**: if comma immediately precedes or follows an adverb or adverbial phrase, e.g., *Probably, that will not work.*
  – **PARENTHETICAL**: if commas surround a phrase that comments on the surrounding text or indicates who is speaking, thinking or observing – the parenthetical text should be deletable without changing the grammaticality of the surrounding text. It should comment on the text but not change its meaning, e.g., a) *Mary, I believe, is twelve feet tall*; b) *Mary, oh yeah I mean her, is twelve feet tall.*
The beauty of automatic replenishment is that the buyer is really the customer. She is telling us what she wants and needs in the future. Quite frankly, of all the buying we do, letting our customer make the choice seems to make the most sense.

—Tom Cole, Chairman and CEO, Federated Logistics and Operations

Our goal is to replace the product on the retail shelf as quickly as possible, because that’s where the consumer buys it.

—Jeff Kernodle, Vice President for Replenishment, VF Corp

Many of the popular accounts of quick response, rapid replenishment, and supply-chain management assume that all parties—consumers, retailers, and suppliers—win as a result of these policies. Consumers have definitely benefited because these practices afford them a greater choice of products at lower average prices. It is safe to say that lean retailers have also come out ahead, given their rapid growth in relation to, and at the expense of, traditional retailers in many different retail channels. But have suppliers benefited from entering into relations with lean retailers? Have such firms improved their competitive position along with the retailers they supply?
Emailed Question: 1

- When creating an NLP system, how do you decide to use manual rules, a statistics-based approach or a hybrid?
  - Short answer: experiment and see what works.

- Most current NLP is really a combination of some sort:
  - Supervised ML includes manual annotation for training of statistical systems

- Understanding and Complexity
  - Simple well-understood tasks work best with statistical approaches.
  - Complex or incompletely understood tasks lend themselves to manual rules
  - Over time, researchers understanding can improve and complex tasks can become tractable for statistical methods, e.g., MT took decades.

- Advantage of Manual Rules:
  - It is usually clear why the rules work (based on observations)

- Advantage of Automatic Systems:
  - You need only find features, you don't have to know if they work, why they work, or how much they work. They can be simple features or complex ones.

- Often manual rules are used as features in automatic systems
Emailed Question 2

• How do systems automatically grade essay questions?
• Commercial testing agencies use automatic systems to save money in grading essays (note the politics):
  – Old: 2 human editors: adjudicator settles disagreements
  – New: 1 human, 1 automatic, 1 adjudicator as before
  – Savings: approx 33%
• Systems collect features about “good” and “bad” essays
  – punctuation, capitalization, word/sentence length, N-grams, etc.
  – ML system trains to distinguish good from bad
  – They report that the systems are as predictive as human graders
  – It may not be clear why the systems predict correctly
Emailed Question 3

• **How does automatic summarization work?**
• **Different formulations of the problem**
  – Different targets of summarization
    • Summary of a collection of documents
    • Summary of a single document
  – Different definitions of what constitutes an answer
    • A cohesive short text that contains the important ideas
      – 50%, 20%, X% of the size
      – one paragraph (abstract)
      – one line (headline)
    • List of snippets of text or list of topics
  – General Summary or Focused on a question/set of questions
Summarization Strategies

• Choose “important” sentences and important words:
  – Use various scores (IDF, centrality, frequent terms, etc.)
  – Focus on sections likely to summarize (initial and final paragraphs)

• Shorten sentences
  – Remove some modifiers (non-negating adverbs, adjectives, ...)
  – Convert sentences to nominalizations

• Do not repeat information
  – Merge information and remove duplicates

• Other techniques based on natural language generation
  – Ordering of sentences in a summary based on Natural Language Generation

• More Info in J & M: Chapter 23
More Questions?

• I am happy to answer more questions.