Intro to Computational Linguistics: Final Review Lecture

Adam Meyers
New York University
Summary

• Administrative Details
• Regular Expressions
  – Writing them applying them to data
• Phrase structure and POS
  – Annotation
  – Algorithms using them or for detecting them: Viterbi, HMM, CKY parsing
• Bigrams and Unigrams
• TFIDF and Cosine similarity
• Sequence Labeling with BIO tags: Noun Groups and NEs
• 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
  – No email, texting, using programs for algorithms
• 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 = ∅
  - The empty set (base case 1, doesn't recognize any strings)
- Regexp = ε
  - The empty string (base case 2, recognizes 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 then 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 \( A \)
Regexp Notation Continued

- Disjunction of characters
  - $[ABC]$ – means the same thing as $A | B | C$
  - $[a-zA-Z0-9]$ – ranges of characters equivalent to listing characters, e.g., a|b|c|...|A|B|...|0|1|...|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
    - $A(BC)|(DEF)$ means ABC or ADEF
    - Otherwise defaults apply, e.g., $ABC|D$ means $ABC$ or $ABD$

- ? signifies optionality
  - $ABC?$ is equivalent to $(ABC)|(AB)$

- + indicates 1 or more
  - $A(BC)*$ is equivalent to $A|(A(BC)+)$
Regexp Notation Continued

• 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)

• What is a “good” regexp to solve some task
  – Not overly specific
    • Capture generalizations
    • Covers some unseen examples
  – Not overly general
    • Should not match obviously wrong cases
Sample Regular Expression

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
- Valid patterns include: Month Day; Month, Year; Month; year
  - A comma and a space precede the year, when it occurs with other elements:
    - Month, Year or Month Day, Year

Bad answers

- Too Specific: \( (\text{January}(3)?, 2012)|(\text{January }3)|(\text{Jan. }3, 2012) \)
- Too General: \[A-Za-z\.-]+\([0-9]\)?,(\[0-9\]+)*\]

Good Answer: \((\text{Jan(uary)?)|(Feb(ruary)?)|(Mar(ch)?)|(Apr(il)?)|(May)|(Jun(e)?)|(Jul(y)?)|(Aug(ust)?)|(Sep(t?)(ember)?)|(Oct(ober)?)|(Nov(ember)?)|(Dec(ember)?))\.(\[1-3]?[0-9]\)?,(\[0-9\]\{4\}))\)
Phrase Structure Rules and Tree

• Draw a Phrase Structure Tree, including Penn POS tags, and List the Rules for the following sentence from Wikipedia (shortened slightly):
  – Parodia tenuicylindrica is a small species of cactus native to Brazil
  – We will assume that:
    • species names consist of proper nouns
    • all the words in proper noun phrases are proper nouns
  – This phrase native to Brazil is the most difficult part of this sentence to analyze and I may ask a specific question about this.
Parodia tenuicylindrica is a small species of cactus native to Brazil
Phrase Structure Rules used

- $S \rightarrow NP\ VP$
- $NP \rightarrow NNP\ NNP$
- $NP \rightarrow NNP$
- $NP \rightarrow NN$
- $NP \rightarrow DT\ ADJ\ NN$
- $NP \rightarrow NP\ PP\ ADJP$
- $PP \rightarrow IN\ NP$
- $VP \rightarrow VBZ\ NP$
- $ADJP \rightarrow JJ\ PP$
Things to Remember about Phrase Structure

• PP $\rightarrow$ P NP
  – *in the room, at the table, by John, with gusto*

• Sbar $\rightarrow$ special_word S
  – special_word $\rightarrow$ that, for, subord_conj, wh_word
  – *that she would leave soon*
  – *for her to leave*
  – *if she leaves*

• CC combines 2 or more Xs to produce a new X
  – *[NP [NP John] [CC and] [NP [DT the] [NN blender]]]*
  – *[S [S The ball went up] [CC or] [S Maybe it didn't]]*

• Punctuation matters – it should be included, typically with itself or PU as its POS
Chomsky Normal Form (required by CKY)

- Context Free Grammars can be converted to CNN
  - 3 types of rules:
    - XP → YP ZP  Binary Branching
    - XP → x  NonTerminal → terminal
    - XP → ε  NonTerminal → 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:
      - $NP \rightarrow NG PP$
      - $NG \rightarrow POSSP N$
Sample Grammar for CKY

- $S \rightarrow NP \ VP$
- $NP \rightarrow D \ N$
- $NP \rightarrow \text{PossP} \ N$
- $NP \rightarrow N$
- $\text{PossP} \rightarrow NP \ Poss$
- $VP \rightarrow V \ NP$
- $VP \rightarrow V$
- $N \rightarrow \text{clam}$
- $N \rightarrow \text{edges}$

- $N \rightarrow \text{shell}$
- $D \rightarrow \text{the}$
- $Poss \rightarrow 's$
- $V \rightarrow \text{has}$
- $V \rightarrow \text{edges}$
- $V \rightarrow \text{shell}$
## CKY Parse Chart

<table>
<thead>
<tr>
<th>The</th>
<th>clam</th>
<th>'s</th>
<th>shell</th>
<th>had</th>
<th>edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>D</td>
<td>NP</td>
<td>POSSP</td>
<td>NP</td>
<td>S</td>
</tr>
<tr>
<td>1</td>
<td>N, NP</td>
<td>POSSP</td>
<td>NP</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>2</td>
<td>POS S</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>N, NP,V,VP</td>
<td>S</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>V, VP</td>
<td>VP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>N, NP,V,VP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### D [0,1]
- NP [0,2]
- POSSP [0,3]
- NP [0,4]
- S [0,5]
- S [0,6]

### N, NP [1,2]
- POSSP [1,3]
- NP [1,4]
- S [1,5]
- S [1,6]

### POSS [2,3]
- N, NP, V, VP [3,4]
- S [3,5]
- S [3,6]

### V, VP [4,5]
- VP [4,6]

### N, NP, V, VP [5,6]
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>Start</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NNP</td>
<td>.42 * .01</td>
<td><em>0</em>.001</td>
<td><em>0</em>.001</td>
<td>* .23* .001</td>
</tr>
<tr>
<td>NNS</td>
<td><em>0</em>.03</td>
<td><em>0</em>.03</td>
<td><em>3</em>.03</td>
<td>* .24* .03</td>
</tr>
<tr>
<td>NN</td>
<td>.20 * .02</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBZ</td>
<td></td>
<td><em>3</em>.05</td>
<td><em>1</em>.05</td>
<td><em>0</em>.05</td>
</tr>
<tr>
<td>VBD</td>
<td>.05 * .05</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>End</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**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>Start</td>
<td>1</td>
<td><em>0</em> .001 = 0</td>
<td><em>0</em> .001 = 0</td>
<td>.23* .001 = 9.67 * 10^-7</td>
</tr>
<tr>
<td>NNP</td>
<td>.42 * .01</td>
<td><em>0</em> .001 = 0</td>
<td><em>0</em> .001 = 0</td>
<td>*0* .001 = 0</td>
</tr>
<tr>
<td>NNS</td>
<td>*0* .03 = 0</td>
<td><em>3</em> .03 = 3.6 * 10^-5</td>
<td>*0* .03 = 1.8 * 10^-5</td>
<td></td>
</tr>
<tr>
<td>NN</td>
<td>.20 * .02</td>
<td>*0* .03 = 0</td>
<td><em>3</em> .03 = 6.3 * 10^-5</td>
<td>*0* .05 = 1.25 * 10^-5</td>
</tr>
<tr>
<td>VBZ</td>
<td>*0* .05 = 0</td>
<td><em>3</em> .05 = 6.3 * 10^-5</td>
<td>*1* .05 = 1.25 * 10^-5</td>
<td></td>
</tr>
<tr>
<td>VBD</td>
<td>*0* .05 = 0</td>
<td>*0* .05 = 0</td>
<td>*0* .05 = 0</td>
<td></td>
</tr>
<tr>
<td>End</td>
<td>*0* .05 = 0</td>
<td>*0* .05 = 0</td>
<td>*0* .05 = 0</td>
<td>*0* .05 = 0</td>
</tr>
</tbody>
</table>

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

- Fill in: max (previous X transition X likelihood)
Unigram & Bigram Language Models

• If a system produces multiple possible sentences, models can be used to determine which sentence is more probable?

• System output of:
  – Machine Translation
  – Speech Recognition
  – Language Generation
  – Automatic Summarization

• Unigram model: product of probabilities of each word independently
  – probability of word = count of word in corpus/total words in corpus

• Bigram model: product of probabilities of each word
  – probability of word = Count(prev_word,word)/Count(prev_word)

• OOV model – different possible OOV models, but we assume:
  – Words in training that occur once are instances of *OOV*
Applying Unigram Model

• Counts of these words in the Brown Corpus using NLTK
  - A 23195 fact 447 about 1815 the 69971 unicorn 0 is 10109 the 69971 same 686
    as 7253 an 3740 alternative 34 fact 447 about 1815 the 69971 unicorn 0. 49346

• Unigram probability—there are 1161192 words in Brown
  - A 0.02 fact 0.000385 about 0.00156 the 0.0603 unicorn 0.0 is 0.00871 the 0.0603
    same 0.000591 as 0.00625 an 0.00322 alternative 2.93e-05 fact 0.000385 about
    0.00156 the 0.0603 unicorn 0.0. 0.0425

• Unigram Probability with OOV model (15673 OOV words)
  - a 0.02 fact 0.000385 about 0.00156 the 0.0603 unicorn 0.0135 is 0.00871 the 0.0603
    same 0.000591 as 0.00625 an 0.00322 alternative 2.93e-05 fact 0.000385 about
    0.00156 the 0.0603 unicorn 0.0135. 0.0425
Applying Bigram Model

- *start_end* a 0.0182
- a fact 0.000388
- fact about 0.00447
- about the 0.182
- the *oov* 0.0293
- *oov* is 0.00485
- is the 0.0786
- the same 0.00898
- same as 0.035

- as an 0.029
- an alternative 0.00241
- alternative fact 0, thus 0.000385 (unigram for fact)
- fact about 0.00447
- about the 0.182
- the *oov* 0.0293
- *oov*. 0.0865
- . *start_end* 1.0
- Total = product of probs = 1.12e-30

- Bigram = freq of seq/freq of first word
- If Bigram = 0, use unigram instead
- See bigram_test.py file
  - http://cs.nyu.edu/courses/fall17/CSCI-UA.0480-006/bigram_test.py
TFIDF

• TFIDF – Property of Term with respect to a document
  – keyword suitability, representativeness of a topic, etc.
  – Uses: Doc Retrieval, Term Extraction, etc.
• TF = frequency in a document
• IDF = number of documents in sample divided by number of documents containing word
• TFIDF = TF * log(IDF)
• Example: “rock” occurs 10 times in document X. It occurs in 100 out of 3000 documents in collection. TFIDF = 10*log(3000/100) = 34.01
Cosine Similarity Between Query and Document

\[ \text{Similarity}(A, B) = \frac{\sum_i a_i \times b_i}{\sqrt{\sum_i a_i^2 \times \sum_i b_i^2}} \]

• Example:
  – the terms in the vectors include: animal, vegetable, mineral, monkey, golf enthusiast
  – The vector for the query is: [0,0,0,34,.8]
  – The vector for a given document is: [1,2,3,4,5]
  – What is the similarity?
    \[ \frac{0\times0+0\times2+(34\times4)+(0.8\times5)}{\sqrt{(0+0+0+34^2+8^2)\times(1^2+2^2+3^2+4^2+5^2)}} = \frac{140}{\sqrt{640\times55}} \approx 0.2014 \]
Sequence Labeling with BIO tags

• Noun group BIO tags
  – The B
  – big I
  – bad I
  – wolf I
  – approached 0
  – the B
  – house I

• NE BIO tags
  – However 0
  – , O
  – International B-ORG
  – Business I-ORG
  – Machines I-ORG
  – and O
  – Google B-ORG
  – rose 0
  – in 0
  – active 0
Basic NE types

• Person – a person name (*Mary Smith*) or a set of people (*the Smith family*)

• GPE – Name associated with land mass, a government and the people who live there (*the United States, New Jersey, …*)

• ORG – Name associated with a company, club, or other type of structured unit with members, employees and/or other types of participants (*IBM, the Catholic Church, the NY Police Department, ..*)
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
  – John likes Mary's drawing of him
  – John likes his drawing of Mary

• Case 2: Otherwise, the antecedent must be outside the immediate tensed clause containing the personal pronoun.
  – John said that he liked pizza.
  – John wanted for him to like pizza.
  – John liked 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:
  – John always carries a slice of pizza with 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 following p, left-to-right, breadth first, but not going below any S or NP node found. Propose each NP.

End
Antecedent Found?
Yes
No
End
Hobbs Search Example

1. Mary saw the chicken.

2. Jim said that she laughed.
Evaluation

- **Accuracy**: If the answer key and system output are guaranteed to be the same length
  
  \[ \text{Accuracy} = \frac{\text{Correct}}{\text{Total Items}} \]

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

- **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:
    
    1. 2 system entities: \( \{A_1, A_2, A_3, A_4, C_1\} \) and \( \{B_1, B_2\} \)
    2. 2 answer key entity: \( \{A_1, A_2, A_3, A_4, B_1, B_2\} \) and \( \{C_1\} \)
    3. Precision = \((4 \times 4/5) + 1/5 + 2\) \(\times\) \(1/7 = .77\)
    4. 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
Answer to Sample FS Question
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

• Initial: *la/the* (.25), *maisson/the* (.25), *la/house* (.25), *maisson/house* (.25)

• *la maison* → *the house*
  – *la/the X maison/house* = .0625
  – *maisson/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/the* = .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 Likelihood 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*
  - *la/the* (.44), *maison/the* (.29), *la/house* (.42), *maisson/house* (.42)
  - *la/the* X *maison/house* = .1848
  - *maisson/the* X *la/house* = .1276

- *la maison bleu* → *the blue house* (all possible alignments)
  - *la/the* X *maison/house* X *bleu/blue* = .06098
  - *la/the* X *maison/blue* X *bleu/house* = .02468
  - *la/house* X *maison/the* X *bleu/blue* = .04019
  - *la/house* X *maison/blue* X *bleu/house* = .02356
  - *la/blue* X *maison/house* X *bleu/the* = .045274
  - *la/blue* X *maison/the* X *bleu/house* = .016269

- *La fleur* → *the flower*
  - *la/the* X *fleur/flower* = .22000
  - *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>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>blue</td>
</tr>
<tr>
<td><em>la</em></td>
<td>.70</td>
</tr>
<tr>
<td><em>maisson</em></td>
<td>.24</td>
</tr>
<tr>
<td><em>bleu</em></td>
<td>.25</td>
</tr>
<tr>
<td><em>fleur</em></td>
<td>.19</td>
</tr>
</tbody>
</table>

- Unigram probabilities (count in WSJ ÷ 1 million)
  - *the* = .035, *blue* = 1.3 X 10⁻⁴, *house* = 6.7 X 10⁻⁴, *flower* = 6 X 10⁻⁶
- The most probable translation would be:
  - *the house blue* = translation-prob X language prob = 4.37 X 10⁻¹⁰
    - translation-prob = .7 X .5 X .41 = .1435
    - Lang-prob = .035 X 6.7 X 10⁻⁴ X 1.3 X 10⁻⁴ = 3.05 X 10⁻⁹
Sample Annotation Task

• Semantic Role Labeling: Find each noun like *ABILITY* (listed in allcaps) and mark 2 arguments: a sentient NP that can do an action (ARG0) and a description of the action or type of action (ARG1). ARG0 and ARG1 are both optional. Arguments can occur anywhere in the sentence, but prefer close arguments.

• Examples:
  – *[the government]'s ABILITY [to pay its bills]*
    ARG0                  ARG1
  – *[the government]'s legal CAPACITY*
    ARG0                  ARG1
Data to Mark Up

- The government's borrowing AUTHORITY declined at midnight on Tuesday.
- Mr. Honecker headed the Poliburo's security APPARATUS.
- The complicated new funding DEVICE will cause more problems than it will solve.
- The First World has for some time had the bad HABIT of smothering other people's economies with this kind of unfocused kindness.
- So do just about all the losses that could be attributed to the sheer INCOMPETENCE of unqualified planners.
Practice Final and Answers

• Practice Test

• Answers